Inference of gene regulatory networks using dynamic Bayesian networks

Yogmatee Roochun

Master of Science
School of Informatics
University of Edinburgh

2016

Abstract

Gene regulation is an essential component of many biological processes. Understanding how a cell controls the level of expression of a gene, and understanding the regulatory relationships between genes, are two important aims of systems biology.

With the advent of high throughput methods for measuring gene expression, a huge amount of transcriptomic data has become available. This data contains important information about genes and proteins, which can be used to better understand the genetic characteristics of diseases and thus enable effective treatments to be developed. Several methods of analysing gene expression data have been devised over the years. They are used to infer gene networks from the data, however, many of the methods do not consider changes in network topology at different time points in the time series data.

The aim of this thesis is to implement a method developed by Thorne for the inference of gene regulatory networks which can vary their structure between different time points while at the same time taking into account the sequential aspect of the data [Thorne and Stumpf, 2012]. This method uses the Hierarchical Dirichlet Process Hidden Markov Model (HDP-HMM) and uses a Dynamic Bayesian Network (DBN) to represent the gene regulatory network at each hidden state. As such, it is abbreviated as the HDP-HMM-DBN method.

HDP-HMM-DBN may be executed as a stand alone method but it can also be used in combination with GABI, an algorithm implemented by the Overton group, which is designed to perform relevance thresholding in networks. Gabi also predicts directionality using information-theory and the properties of the undirected relevance network. We demonstrate how the use of GABI as a prior to the Hierarchical Dirichlet Process Hidden Markov Model algorithm improves overall performance. The efficiency of the HDP-HMM-DBN method, with and without GABI, is evaluated using benchmark data available on the DREAM challenge website. We focus on the DREAM4 challenge; the provided in silico time-series data is input to our method and the results are evaluated with the the gold standard networks.

To verify the benchmark evaluation we make use of ROC curves. The ROCR package is used to plot graphs for the Matthews correlation coefficient of the results, produced by running the method on its own and with GABI as a prior. We demonstrate how performance is affected when GABI is applied. Parameters for the method were set using cross-validation with DREAM gold-standard data.

As a more real life application of the HDP-HMM-DBN method, we use it to analyse renal cancer time course data provided by the Overton group. This data is derived from the drug resistant Caki-1 cell line which has been exposed to the drug Sunitinib in hypoxia. Cytoscape is used to visualise and analyse the network produced by our method from the cancer data. DAVID (Database for Annotation, Visualization and Integrated Discovery) is applied to gain insights into the biological meaning of the gene network

The biomedical context of this project is to develop more effective clinical tools for renal cancer medicine by investigating molecular control of drug (Sunitinib) resistance and response.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Yogmatee Roochun*)

Acknowledgements

I would like to thank my two supervisors, Thomas Thorne and Ian Overton. Their help and support throughout this project have been invaluable. Thanks also go to Hans-Joachim Sonntag for providing me with renal cancer data and Erola Pairo-Castineira for helping me generate the renal cancer network using Cytoscape.

I am grateful to my employer, the MRC Human Genetics Unit, for sponsoring my masters degree through a staff scholarship. In particular, Richard Baldock, Simon Harding and the GUDMAP team, were supportive during my studies. I would also like to thank Nick Burton and Sophie Marion de Proce for useful suggestions in improving my thesis. Finally, a thought goes to my family for their support throughout my studies.

Table of Contents

List of Figures

Chapter 1

Introduction

Cellular phenotype is the term used to describe the groups of multiple and different cellular processes that take place as part of a cell's proper function [Sul et al., 2009]. These cellular processes combine the activity of thousands of genes, maintained by a complex network which controls their expression. Understanding this organisation of genes and the interactions between them is an important part of systems biology as it sheds light on the normal physiology of the cell. An important part of systems biology is the inference of regulatory networks from the gene expression data obtained from high-throughput experiments such as microarrays.

Genome projects are providing us with a huge amount of clinically relevant information in the form of genes and proteins that deal with cellular function. By analysing this information we get better insight into the complex molecular processes essential to many biological systems. Experimental techniques on their own are not sufficient to allow us to fully comprehend the complexity of genetic networks. Recently there has been substantial progress in the application of computational techniques to the field of biology [Imoto et al., 2003; Lebre et al., 2010; Werhli and Husmeier, 2008; Penfold et al., 2012; Thorne and Stumpf, 2012], the main goal being the deciphering of complex functional relationships occurring between genes by analysis of omics data. One simple method of investigating the relationships between different expression datasets, and understanding their functional pathways, is by performing clustering of gene expression profiles.

Over the last decade remarkable progress has been made in genomic research. This has led to an increase in the number of reverse engineering methodologies proposed in the literature. The main objective of these methods is to infer, analyse and understand the causal relationships between genes: in other words to understand which gene is the regulator to another [DâĂŹhaeseleer et al., 2000]. Reverse engineering involves taking gene expression datasets, deciphering the information contained in them and using that information to unravel unknown gene regulatory networks. There are several ways of regenerating gene regulatory networks, based on different paradigms, for instance: using graphical Gaussian models [Schäfer and Strimmer, 2005], Bayesian networks [Friedman et al., 2000], Boolean networks [Lähdesmäki et al., 2003] and so on. A lot of these methodologies do not account for the varying structure of the network at different

time points. Those which do often do not take into consideration the sequential nature of the data. Identifiability is another important feature a reverse engineering method should satisfy to enable more accurate predictions. However, it is often an overlooked property when modelling gene regulatory networks [ide, 2016].

The HDP-HMM-DBN method implemented in this thesis performs inference of gene networks while taking into consideration the changing network topology at various time points in the data. Furthermore, by using it in combination with the GABI algorithm to prune out insignificant edges, its performance is significantly improved.

The efficiency of our method is verified using a rigid evaluation process involving benchmarking and cross-validation using data from the DREAM[1] challenge website (Chapter 5). We run our method on renal cancer data, derived from cell lines which have been exposed to the drug Sunitinib, and analyse the results produced using specific programs such as Cytoscape[2].

From a biological point of view, this method is a valuable tool which can be applied to gene expression data to infer the complex web of genes contained therein, and their regulatory relationships. By understanding the link between the regulators and their targets one can understand the genetic characteristics behind diseases, and as a result find cures for them.

The model that we will be implementing for this project is based on Dynamic Bayesian networks and Hierarchical Dirichlet Process-Hidden Markov Model. More details on that are given in the chapters which follow.

1.1 Motivation

A gene regulatory network describes a collection of genes that interact with each other in order to allow for the proper functioning of a particular cell. The way the genes are expressed is quite specific in the sense that each one needs to be expressed in the correct amount and at the right time to ensure the correct behaviour of the cell.

Analysing the complex structure of gene regulatory networks using computational methods is an important aspect of systems biology. Gene expression data sets tend to be high dimensional because of the large number (1000s) of expressed genes in a given cell. The data also contains relatively few sampling time points in contrast to its high number of dimensions. For this reason, it is called the large p (number of genes) and small n (number of time points/samples) problem, which makes it difficult to analyse.

To date, various mathematical models and computational methods have been developed to infer gene regulatory networks from gene expression profiles [Schäfer et al., 2001; Wang et al., 2016; Lebre et al., 2010; Grzegorczyk et al., 2008]. However a lot of these network models assume that the network topology stays the same over time

[1]DREAM Website: http://dreamchallenges.org/

[2]Cytoscape: http://www.cytoscape.org/

when in reality this is often not the case. Some approaches have been proposed which build time-varying biological networks [Lebre et al., 2010]. These approaches make use of change points in the time series data, but fail to take into account the sequential nature of the data . Another issue is that they establish in advance the number of change points that can be observed, and are therefore unable to adapt to the complexity of the observed data.

To allow for the inference of gene regulatory networks which can change their structure at different time points, whilst taking into consideration the sequential nature of data, Thorne and Stumpf [2012] has proposed a methodology which uses the infinite Hidden Markov Model, also known as the *Hierachical Dirichlet Process Hidden Markov Model*, along with a Bayesian network representing the gene regulatory network at each hidden state. The Hierachical Dirichlet Process Hidden Markov Model (HDP-HMM) extends the traditional hidden Markov model by having an infinite number of hidden states to explain the data. The distributions of the states are grouped in a hierarchical structure to facilitate sharing and transitions between the states. The method employs the Metropolis Hasting algorithm when generating network structures for each hidden state. More information about the method is given in chapter 3.

To improve the efficiency of the HDP-HMM-DBN method at making predictions we have integrated it with GABI, a relevance thresholding algorithm developed by Alex Lubbock from the Overton group as part of his PhD. The GABI algorithm generates a relevance network and applies information-theoretic and topological directionality inference rules. When using GABI as a prior to our method, edges with low significance are removed, thereby ensuring that the networks generated by the HDP-HMM-DBN method only contain important edges.

In the next section we give an overview of the aims and objectives of this thesis.

1.2 Aims and Objectives

There are three main aims of this project: implementing the HDP-HMM-DBN method, evaluation and benchmarking the method using data from the DREAM[3] challenge while assessing the benefit of including GABI as a prior and using of our method to analyse renal cancer data provided by the Overton group.

The first task is an implementation of the non-parametric framework proposed in [Thorne and Stumpf, 2012]. This framework, which makes use of the Hierachical Dirichlet Process Hidden Markov Model (HDP-HMM), models how a network changes its topology at different time points. In the original method devised by Thorne et al., the network structure was represented in the form of a Bayesian network. In our implementation we extend it in a novel way by using a dynamic Bayesian network to model the network structure. The advantage of using a dynamic Bayesian network is that it increases the space of potential network structures by overcoming the restriction imposed by the directed acyclic graph (DAG) structure of a Bayesian network. This

[3]DREAM Challenge: www.dreamchallenges.org

method is developed using the R language because of the large number of statistical features it contains.

In the second task, the method is evaluated using benchmarking data from the DREAM challenge website. DREAM[4], (Dialogue for Reverse Engineering Assessment and Methods), is a repository of challenges which allows researchers to evaluate their methodologies against some gold standard. The data that we use to test our method comes from the DREAM4 challenge and is simulated time-course data. We run our method both on its own and in conjunction with GABI. By incorporating a connectivity prior from GABI, we verify whether this enables the HDP-HMM-DBN algorithm to make more accurate predictions.

To determine a thresholding parameter and to benchmark the various instances of the network inference algorithm, we compare the output of the HDP-HMM-DBN algorithm against the provided gold standard networks. Receiver operating characteristic (ROC) curves are used to verify the performance of our method. We also vary the *"number of node (gene) parents"* parameter when running the method with and without GABI in order to verify how this impacts the method's outcome. We generate Matthews Correlation Coefficient curves for the various results, corresponding to different runs of the method, and use them to evaluate how well our method executes when the settings are changed. Prior to running the algorithm on the renal cancer data, we perform cross-validation of the method to find the settings which give optimal performance. We then apply these settings to our method when running it on the cancer data.

Finally in the third task, we run our method on renal cancer time course transcriptome data derived from cell lines exposed to the drug Sunitinib. Using the output of our method, we visualise regulatory networks using Cytoscape and use them to make further biological investigations.

We put forward a checklist which may be used to verify whether we have completed the tasks that have been set at the start of the project. This will give us an indication of the project's success.

1. *HDP-HMM framework implementation*
    - Implement the method developed by Thorne and Stumpf [2012] and extend it in a novel way by exploring the inference of gene regulatory networks using the Dynamic Bayesian network approach.
    - Incorporate GABI as a prior in the method.
2. *Benchmarking using DREAM data*
    - Assess the performance of the method using DREAM networks as benchmark.
    - Verify how using the method in combination with GABI affects performance.

[4]DREAM Website: http://dreamchallenges.org/

- Assess the effect of the maximum number of potential parents of a gene parameter on predictive performance, which is limited by the availability of the number of replicates in the dataset.

3. *Running the method on renal cancer data*
   - Run the method on the renal cancer data provided and generate a molecular network which can be used for further biological analysis.

1.3 Results and Contributions

The main contributions of this thesis are as follows:

1. *Novel method for inferring gene regulatory networks from gene expression data.*
   We implement the method proposed in [Thorne and Stumpf, 2012] and extend it in a novel way by using a dynamic Bayesian network to represent the gene networks.
2. *Incorporating a connectivity prior.*
   We integrate GABI with our method to perform relevance thresholding by removing insignificant edges in the networks generated by our method. We show in particular how using a connectivity prior helps achieve better results.
3. *Performing dynamic analysis of renal cancer data.*
   By running our method on renal cancer time course data, we are able to look at the gene networks at specific time points during the gene expression phase. For example, we can study the gene networks during apoptosis and angiogenesis. By being able to perform such in-depth analysis of biological data, we can better understand the interactions between the regulatory elements when they are exposed to specific conditions like hypoxia or normoxia. We can also analyse the effect of certain drugs on the cancer cells. This is an important step when trying to find cures for specific diseases.

1.4 Thesis Structure

The content of this thesis is structured in the following chapters.

Chapter 1: *Introduction* gives an overview of the method we have implemented and describes the motivation for the project as well as our aims and objectives. It also lists the main contributions it has brought to research.

Chapter 2: The *Background* chapter provides background details about the molecular biology aspect of the project by explaining concepts like the central dogma, gene expression and gene regulatory networks. It investigates the various existing mathematical models and computational methods that have been developed to infer gene regulatory networks from gene expression profiles. It also provides information on

some important concepts related to the *Hierarchical Dirichlet Process-Hidden Markov Model* framework. An entire section is dedicated to explaining the biology of cancer as this is an important part of this thesis.

A lot of the research work in this chapter is derived from the research proposal which was written as part of the *Informatics Research Proposal* module

Chapter 3: The *Algorithm* chapter describes the *Hierarchical Dirichlet Process-Hidden Markov Model* and the *Metropolis Hastings sampler* algorithms in more details. Pseudo-codes for both of them are also listed.

Chapter 4: The *Implementation* chapter describes how *HDP-HMM-DBN* method has been developed in R. Some code extracts and brief descriptions of the functions which are implemented as part of the method are also included.

Chapter 5: *Benchmarking and Analysis* looks into the DREAM challenge and provides a step by step description of how the HDP-HMM-DBN method is evaluated using simulated time series data from the DREAM4 challenge. Here we talk about the algorithm's performance in terms of the predictions made. We discuss the results of running the HDP-HMM-DBN algorithm both on its own and when incorporating GABI as a prior. We vary the parameter of the *number of parents associated to a node* and run the method both with and without GABI. We present ROC curves and Matthews correlation coefficient graphs for the different runs of the method as part of the evaluation process. We also perform cross-validation of the results and present it in the form of tables. This chapter also covers the analysis of renal cancer data using the HDP-HMM-DBN method and describes the steps undertaken to generate a molecular network using the method's output. We give an overview of the biological interpretation of the network by highlighting the important relationships between specific genes which are known to be linked to cancer.

Chapter 6: *Conclusion* reflects on what has been accomplished in this project, our observations and suggestions for future works.

Appendices: List of all graphs and tables generated as part of the evaluation process and any other additional materials referred in the main text.

Chapter 2

Background

This chapter provides the biological and computational background of this thesis. It gives an overview of the central dogma of molecular biology and the three elements which form part of it namely DNA, RNA and protein. It explains the process of how genetic information in a cell gets translated into protein. It also covers an exposition of the relevant literature review on gene expression, how it is measured, analysed and modelled as well as the various methodologies devised for the reverse engineering of gene regulatory networks from gene expression data. Some background information on concepts like Markov chain, Hidden Markov model, Dirichlet process and Hierarchical Dirichlet process are also given as these are important aspects of the *Hierarchical Dirichlet Hidden Markov Model* algorithm implemented as part of this thesis.

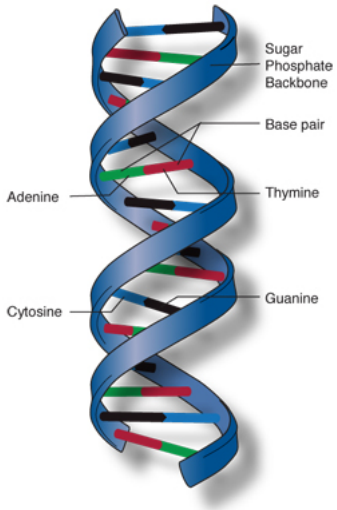
2.1 The Central Dogma of Molecular Biology

The *Central Dogma of molecular biology*, proposed by Francis Crick in 1958 [Crick et al., 1970], is a concept which details how genetic information found in DNA gets converted to protein through processes known as transcription and translation. Before going into more details about the central dogma, we first need to understand what a DNA is.

2.1.1 DNA

DNA which stands for DeoxyriboNucleic Acid is the material which stores genetic information that gets passed on from generation to generation, thereby allowing the reproduction of living things. All organisms inherit from their parents the genetic information which defines their structure and function [Bruce Alberts, 2002]. The DNA acts as a reservoir of genetic information, necessary for the creation and maintenance of an organism. The information found in DNA is composed of four different bases: A(adenine), T(thymine), C(cytosine), G(guanine) and is stored as a code. A DNA code can be very long, for example the human DNA is made up of about three billion bases.

Approximately 99% of these bases are the same in all people [dna, 2016]. The ordering of bases is important as it is this sequence which contains the information required for making proteins. Each base forms a nucleotide by attaching to a sugar molecule and a phosphate molecule. A sequence of DNA bases can pair up with another complementary sequence and in doing so Adenine forms a base pair with Thymine and Cytosine forms a base pair with Guanine. This feature of base-pairing enables two complementary DNA strands to form a double helix.



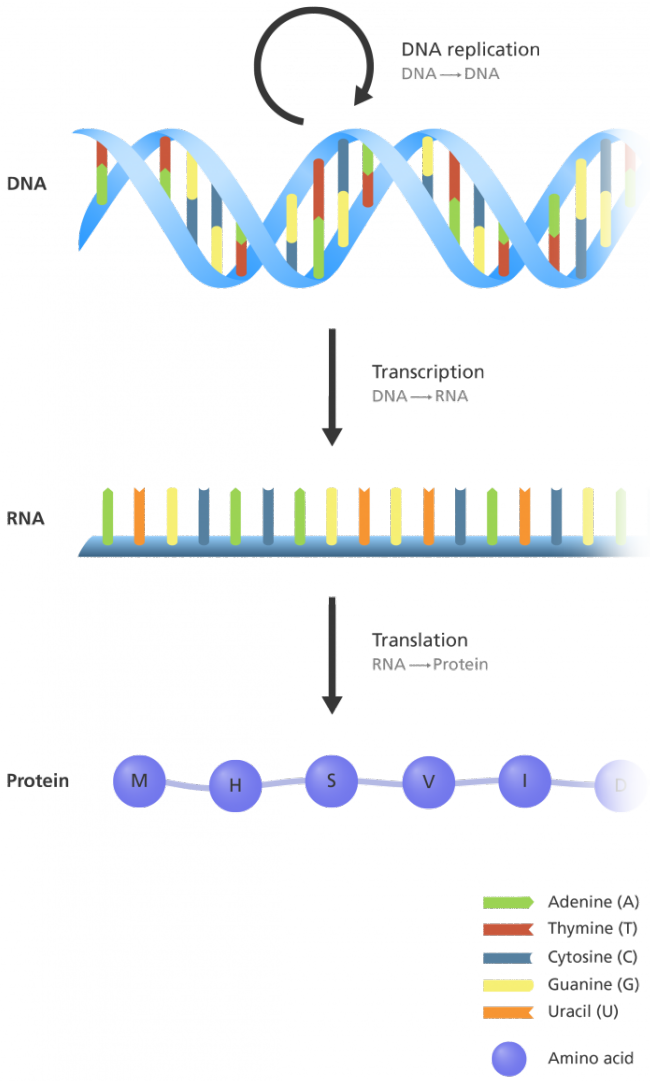
Source: GeneEd (https://geneed.nlm.nih.gov)

Figure 2.1: Double helix structure of a DNA molecule

A DNA sequence consists of multiple genes. Genes are fundamental to heredity in that they are transmitted from an organism to its offspring and are responsible for that offspring's inherited features. A gene may encode for information that directs the manufacture of a specific protein or RNA molecular form. Genetic information flows from the DNA, which acts as the information store, through RNA molecules where the information is translated into proteins. Proteins are the main working components of organisms, playing a major role in almost all the key processes of life.

2.1.2 DNA, RNA and Protein

The key relationship between DNA, RNA and proteins is represented by the central dogma of molecular biology, which explains how genetic information flows from the DNA through RNA molecules and is subsequently used in the formation of proteins.



Source: yourgenome (http://www.yourgenome.org/)

Figure 2.2: Central Dogma of Molecular Biology

The DNA sequence is decoded in a two stage process, the stages being called transcription and translation.

Transcription is the first step of gene expression. During this stage a particular segment of DNA is copied into ribonucleic acid (RNA). This process is carried out by the enzyme RNA polymerase. RNA molecules are linear polymers made up of four bases: Adenine, Guanine, Cytosine and Uracil. Unlike DNA, which has Thymine as one of its bases, RNA contains Uracil instead and is much shorter in length when compared to a DNA molecule. While DNA carries information about many proteins, RNA mainly carries information for a single protein. Messenger RNA (mRNA) is the term used for RNA transcribed from a protein-coding gene and is the molecule that directs the synthesis of the protein chain. A gene is said to be expressed if, when transcribed, it results in an RNA. The expression of genes in a cell can be regulated by the cell itself

by controlling the transcription process which in turn regulates the production of RNA.

Translation is the second step in gene expression where mRNA produced during the transcription phase is converted into protein by a ribosome. Messenger RNA is translated into protein according to the genetic code, where each set of three consecutive bases in the mRNA forms a codon which in turn specifies a particular amino acid. The sequence of nucleotides of a gene gets translated into a sequence of amino acids which forms part of a protein chain having a particular function.

Proteins are the end-product of the translation step. They are manufactured using the information encoded in DNA and are the main working components in organisms, playing a major role in almost all the important processes of life. Proteins are molecules that carry out processes such as energy metabolism, biosynthesis and intercellular communication. Each type of protein consists of a precise sequence of amino acids. Proteins cannot do much on their own. The biological properties of a protein molecule are defined by its physical interaction with other molecules, which may be other proteins or regulators. By interacting with other molecules, proteins form brief or stable complexes which enable them to carry out their function and activity. For example, antibodies attach to viruses or bacteria to mask them for destruction, and actin molecules bind to each other to assemble into actin filaments [ncb, 2016].

Protein-protein interactions occur when a protein physically bind with one or more other proteins in order to perform a particular task. Depending on the protein's function, the binding can be very tight and lasting or weak and transient.

Molecular interactions are important because they help us to understand a protein's function and behaviour. They can help predict the biological processes that a protein of unknown function is involved in. For example we may deduce the as yet unknown function of a protein if it is associated with one of known function. One way of finding out the associations is by looking at the protein networks. Proteins with similar functions, or which are involved in the same process, are normally clustered together in network maps. This knowledge helps in the identification of protein complexes and pathways in networks. [ebi, 2016].

2.2 Gene Expression

In any given cell, thousands of genes are expressed and work together to ensure its proper functioning. Nearly every cell found in an organism is composed of the same set of genes. However only a small number of them are "turned on" (expressed) at any given time. The gene expression of a particular cell is what differentiates it from other cells. For example the gene expression of a hair cell is different from that of an eye cell. Similarly the gene expression of a normal healthy cell is different from that of an abnormal cell such as a cancer cell. The way the genes are expressed is quite specific. For a normal healthy cell, each gene needs to be expressed in the correct amount and at the right time to ensure the correct behaviour of the cell.

Gene regulation is the term used to represent a set of cellular processes, the two main

ones being transcription and translation. These processes are involved in controlling the level of a gene's expression; resulting in the production of a specific quantity of target protein [Filkov, 2005].

A gene regulation system is made up of genes, cis-elements, and regulators called transcription factors. Transcription factors are proteins that bind to specific sequences of DNA to regulate the transcription process of converting DNA to RNA. Interaction between transcription factors and cis-elements during transcription determines the degree of gene expression in a cell and this forms *gene regulatory networks*. Transcription factors act as inducers and suppressors by activating or inhibiting the expression of a gene.

Regulation of gene expression is performed by cells to control the amount of RNA, and thus the amount of protein produced. The amounts and types of mRNA molecules produced in a cell during the transcription phase define the function of that cell as it is these mRNA transcripts that get translated into protein, the cell's functional product. Since a single mRNA molecule can code for several proteins, the control point for gene expression is usually assumed to be at the start of the transcription phase [gen, 2016b].

Gene regulation also controls *cell differentiation* whereby generic embryonic cells are transformed into cells that are specialised for a particular function. For example, a sperm cell is different from a liver cell in both structure and activity performed [cel, 2016a]. Cell differentiation occurs during the gene expression process. As part of this specialisation phase, the cell changes its size and structure as well as the way it responds to signaling molecules whose task is to inform the cell of its function.

2.2.1 Gene Expression Analysis: Methods and Techniques

To derive meaningful information from gene expression data, each gene is studied under multiple conditions and their expression over a certain time span is documented. These time series datasets are then analysed in depth to get insights into normal cellular functions such as differentiation, and to understand the genetic aspect of diseases. Gene expression analysis can be done at any point during the processes of transcription and translation. For instance, the analysis may be performed during or after transcription, or during or after translation. It is common however in gene expression analysis experiments to study transcriptional regulation processes [gen, 2016a]. An indication of how active a gene is can be obtained by the amount of mRNA produced during transcription [Filkov, 2005].

Gene expression is generally assessed by measuring how much mRNA has been produced in a tissue sample during the transcription phase, using methods like: Northern Blot, RNA sequencing (RNA-Seq), reverse transcription polymerase chain reaction (RT-PCR) and microarray analysis. Protein concentrations can also be measured by directly measuring protein levels using a technique known as the Western Blot. A brief description of each of these methods ensues.

2.2.1.1 Northern Blot

Northern Blot is a method used to measure the amount of RNA expression of genes in a particular tissue sample. The first step in this method involves separating the RNA into separate strands according to their sizes using gel electrophoresis. The RNA is next transferred onto a special blotting membrane. This membrane is treated with a probe which is a small piece of RNA. The probe binds to its complementary RNA sequence on the membrane by forming base pairs. This probe has a label which allows the RNA molecule of interest to be detected by simply finding the location of that probe using its label.

2.2.1.2 RNA-Seq

RNA-Seq is a novel method which has been developed for transcriptome profiling. A transcriptome is the entire set of mRNA transcripts produced in a cell under some specific condition during transcription. RNA-Seq uses deep-sequencing technologies and provides a more accurate measure of the mRNA levels than any other existing method [Wang et al., 2009]. Using this technique, the RNA is isolated and purified before getting converted into a set of cDNA. Each cDNA fragment has sequencing adaptors attached to one or both ends of it. The cDNAs are next sequenced using a sequencing platform. The resulting reads are then analysed by either being aligned to an existing reference genome or assembled de novo. What makes RNA-Seq attractive is the fact that it can be used to detect transcripts belonging to genomic sequences that have not been completely determined. Additionally, unlike microarrays, it has very low background noise [Wang et al., 2009].

2.2.1.3 Western Blot

Western Blot is a method which allows the identification of specific proteins from a complex mixture of proteins derived from a particular tissue or cell. The first step in this method involves mixing the protein sample with a detergent to make the proteins unfold into linear chains. The protein molecules are then separated, based on their molecular weight, using gel electrophoresis. The separated proteins are then transferred to a blotting membrane which gets treated with antibodies having labels known as primary antibodies. These labelled antibodies bind to the proteins of interest and any unbound antibodies are washed away. The membrane is then treated with a secondary antibody which binds to the primary antibody allowing the protein of interest to be detected [wes, 2016].

2.2.1.4 RT-PCR

Reverse Transcription Polymerase Chain Reaction is a versatile method used for the detection and quantification of mRNA levels in a given sample. mRNA levels are measured by performing reverse transcription of the RNA to cDNA. The cDNA is then

amplified using PCR. PCR, which stands for polymerase chain reaction, is a technique used for the amplification of specific DNA fragments. The quantity of each specific target is measured by the strength of the signal emitted from the DNA-binding dyes or probes. This amplification process is done several times, during which measurements of mRNA levels are collected. This method can be used, not only for gene expression profiling, but also for finding out mutations and DNA modifications as well as confirming results derived from microarray analysis [gen, 2016a]

2.2.1.5 Microarray Analysis

Microarrays are used to measure the expression of genes in a cell or set of cells. DNA microarray analysis enables an experiment to be performed simultaneously on thousands of genes in order to measure the expression level associated with them. A DNA microarray, also known as a DNA chip, helps identify the amount of mRNA transcripts present in the cell during transcription, and based on this amount we get an approximate measure of the level of expression of that gene [mic, 2016b].

A typical microarray consists of a surface on which probes are fixed at specific locations called spots. One way of measuring gene expression is to compare the expression of a set of genes from a cell in a particular condition (e.g condition A) to the same set of genes from a reference cell maintained under normal conditions (condition B) [mic, 2016a]. RNA is first extracted from the subject cells and transcribed to cDNA where some of its molecules are substituted with nucleotides labelled with different fluorescent dyes. For example cells in condition A are labelled with red dye while those in condition B are labelled with a green dye [mic, 2016a]. These samples undergo a hybridization process. Each spot on the microarray is bound to a certain amount of cDNA proportional to the level of gene expression represented by the probe [mic, 2016b]. The microarray is then scanned by a laser light which detects the amount of fluorescent dye emitted by the RNA molecules. The amount of fluorescence produced is proportional to the quantity of RNA molecules. The end result of this experiment is an image of the microarray with each spot corresponding to a gene which has an associated fluorescence value representing its expression level.

Recent advances made in the methods used for measuring gene expression which allow thousands of genes to be analysed simultaneously, means that studying time series data is now more feasible and as a result more relevant studies can be made in the field of genomics when querying dynamic biological processes [Bar-Joseph et al., 2012].

Techniques such as cluster analysis and correlation are often employed when studying how the sequence of gene expression changes over the course of time. Clustering methods for example hierarchical clustering have been widely applied to time series data. One important aspect of time series data is the ability to infer causal relationships between genes by investigating the changes in the gene expression.

Several methods, such as hidden Markov models have specifically been developed for time series data [Ghahramani, 2001]. One methodology which goes beyond cluster analysis is the inference of gene regulatory networks from the expression data. The

next section provides more insight on gene regulatory networks and how they are modelled.

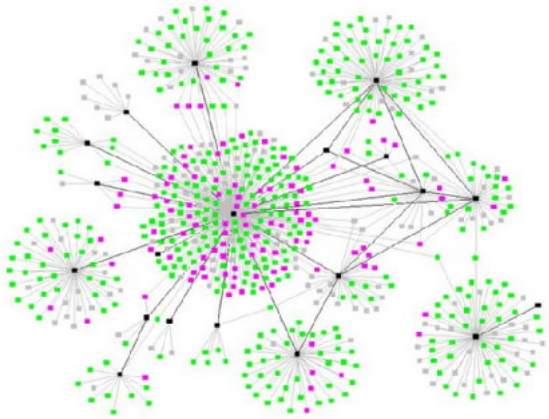
Elements of Sections 2.3 and 2.5.1 extend the materials presented in the Informatics Research Proposal (IRP).

2.3 Gene Expression Modelling: Gene Regulatory Networks

Most biological processes are constantly changing. To capture and study the time-varying aspect of these processes, time-series experiments are performed to measure the gene expression at diverse time points, thereby capturing any transient changes in the expression. Such data is key when modelling the dynamic aspect of biological processes and provides a wealth of varied information, including dynamic cell activity. For instance this data can be used to help understand the relationships between genes by studying their interactions, and can help to identify how genes are expressed and regulated in cellular processes as well as understanding their causal effects.

One way of modelling the dynamic systems in a cell is to have a blueprint which shows the layout of the genetic components, such as genes and proteins, and the interactions occurring between them. Such a blueprint will assist in understanding how genes function cooperatively by their interactions with each other. Gene networks, an example of such a blueprint, concisely represent the complex network of genes in the system being studied.

A **gene regulatory network** is a collection of genes interacting with each other in order to ensure that a cell functions correctly and is fit for its purpose. Understanding the processes behind the proper functioning of these networks is important as they provide insight into the mechanisms by which dysregulations in cellular processes can trigger diseases. It also helps in exploring the effect of drugs in cells, for example by analysing how the gene regulatory network evolves over time when a specific drug is applied to cancer cells. Such knowledge can help produce cures for specific diseases. Gene regulatory networks clearly and comprehensively represent the causality of the interactions between the genes and hence of the developmental processes. They explain exactly how genomic sequence encodes the regulation of expression of the sets of genes that progressively generate developmental patterns and execute the construction of multiple states of differentiation. With advances made in the field of biotechnology, research on gene networks has progressed significantly over the last decade and as a result our knowledge on this subject has also substantially broadened, allowing researchers to effectively model gene regulatory networks.



Source: University of Warwick (http://www2.warwick.ac.uk/)

Figure 2.3: An example of a gene regulatory network with hubs represented by black squares

A gene regulatory network is often depicted as a graph, where each node represents a particular gene and edges represent the potential regulatory relationships between pairs of genes. The relationships between genes can be directed, weighted or signed. A directed edge from node i to node j, denoted by (i, j), means that gene i influences gene j. The weight associated with an edge indicates the strength of the relationship between the two nodes. The sign indicates whether the relationship is an activation: where transcription of other genes is induced, or an inhibition: which is the prevention of transcriptional activity [Cho et al., 2007].

Gene regulatory factors, (for example, transcription factors and their interactions and targets), are very important for the proper functioning of cells. Deciphering the regulatory network structure is crucial to understanding how cellular systems work. Gene networks concisely represent the knowledge of the system being studied and can be used for studying the regulatory interactions between genes during the different stages of organism development.

Knowing how genes interact can help identify the effect of drugs on specific targets. Gene regulatory networks may be used to aid drug development: by predicting adverse effects of new drugs and identifying target genes for the development of new drugs. They may also be used for diagnostic purposes[Nakajima and Akutsu, 2014]. Such knowledge in addition to understanding the behaviour of the model can help with producing cures for certain diseases. Having gene networks of an organism at different time points enable their comparison and provides an understanding of how the network evolves functionally and structurally over time. Differences in the networks provide an insight into how organisms are affected by certain factors or stimuli.

Analysing the complex structure of gene regulatory networks using computational methods is an important aspect of *systems biology*. A feature of gene expression data sets that makes them difficult to analyse is that they have relatively few sampling time points and are highly dimensional (large p small n problem). A number of mathematical models and computational methods have been developed to infer gene regulatory networks from large-scale gene expression profiles. Gene network models can be

classed as either static or dynamic. Dynamic models of gene networks take into consideration the time-varying aspect of changes occurring in the gene expression. They aim to make predictions based on the observed data. Static models of gene regulatory networks do not consider the time component of data, they only display the genes and relationships between them. Some models used to reverse engineer gene regulatory networks include *Boolean networks* [Liang et al., 1998], *Bayesian networks* [Imoto et al., 2003], *Dynamic Bayesian networks* [Thorne and Stumpf, 2012] [Murphy et al., 1999] and *Gaussian Graphical models* [Ma et al., 2007]. A brief explanation of these methods is given below.

2.3.1 Boolean Networks

Boolean networks are examples of dynamic models of gene networks. In this type of network boolean logic is used to depict the state of each node. A variable representing a gene can take on only two values: True or False, represented by 1 and 0 respectively. A value of 1 means that the gene is active and 0 represents an inactive gene. Gene regulation rules are given as Boolean functions with the variables connected by logic operators.

A Boolean network is a directed graph with the nodes represented as Boolean variables. The state of the network corresponds to the combination of values of all the nodes in it. When representing gene networks, the nodes are associated with the levels of gene expression. They indicate whether the mRNA level has gone up or down [Filkov, 2005]. An important assumption of this model is that the genes change state synchronously and do so at discrete time points[Nakajima and Akutsu, 2014]. In other words, the nodes change state at the same time and the network is said to undergo a *state transition* from state $S(t)$ to a new network state $S(t+1)$. The dynamic aspect of Boolean networks and the simplicity they exhibit makes them appealing for use in modelling biological networks.

2.3.2 Graphical Gaussian Networks

Graphical Gaussian models are frequently used when studying gene networks. Correlations present in gene expression data assist in the understanding of the underlying gene regulatory networks. By measuring the amount of independence between a pair of genes using partial correlation, co-regulation patterns occurring between pairs of genes can be inferred, subject to the influences of other genes. This helps differentiate the interactions by classifying them as either direct or indirect [Hache et al., 2009].

Graphical Gaussian models have proven to be useful tools for the inference of gene networks because of the way they model the conditional dependence among the genes. In these models, we assume a random vector denoted by $Y = (Y_1, ..., Y_p)$ following a multivariate normal distribution. Each model is depicted as an undirected graph where the variables Y_j and Y_k are conditionally independent for each non-existing edge (j,k), subject to the remaining variables [Finegold and Drton, 2011].

2.3.3 Bayesian Networks

Bayesian networks, also known as belief networks, are an integral part of the family of probabilistic graphical models (GMs). These models can handle both static and non-static data.

A joint probability distribution can easily be represented by a Bayesian network, so Bayesian networks are often used for the analysis of gene expression data. Using this method, conditional probabilities can be used to represent the gene regulation rules in a gene regulatory network.

A Bayesian network model is made up of two components, a set of nodes (vertices) and a set of directed edges. Each node in the graph represents a random variable (discrete or continuous) and is usually labelled by a variable name as a way of distinguishing it from other nodes. In a Bayesian model of a gene network, a node can represent mRNA concentrations, protein concentrations, genes or other gene regulatory elements [Hartemink et al., 2001]. Relationships between the variables in a Bayesian network may be described as qualitative and quantitative. At a qualitative level, the relationships between the variables are defined by dependence and conditional independence between the nodes, whilst at a quantitative level, the relationships between the variables are described by joint probability distributions whereby the conditional probability distribution at each node depends only on its parents [Hartemink et al., 2001].

The relationship between two nodes *A* and *B*, which represent the probabilistic dependencies between their variables, is denoted by a directed arc between them. The direction on the arrow gives an indication of which node is the precursor to the other one. For example, if there is a directed edge from *A* to *B*, it can be said that the value of variable *B* depends on the variable *A* and node *A* is considered to be the parent of node *B* and *B* is said to be the child of *A*.

The childen, grandchildren, and so on of a node are known as its descendants. A directed path from node *A* to node *B* is depicted as a sequence of edges or nodes which start from A and finishes at B such that each node in the sequence goes in the same direction towards the end node *B*. Each node is a child of the previous node in the path.

The outdegree of a node *n* is the number of edges pointing outward of it. That is the number of children of that node. The indegree of a node *n* is the number of edges pointing towards it. That is the number of parents of that node.

A more formal definition of a Bayesian network, as described in [Friedman et al., 1997] is given below:

A Bayesian network is represented by an annotated directed graph that represents a joint probability distribution over a set of random variables U. *The network is represented by* B=(G,Θ)*, where* G *is a directed acyclic graph consisting of vertices* X_1,...X_n *and edges. The vertices denote random variables, while the edges represent the direct dependencies between them. Each variable* X_i *in the graph* G*, given its parents, is not affected in any way by its non-descendants.* Θ *denotes the set of parameters of the*

network and is denoted by $\theta_{X_i \mid \pi_{X_i}} = P_B(X_i \mid \Pi_{X_i})$ *for each value* x_i *of* X_i *and* π_{X_i} *of* Π_{X_i}*, where the set of parents of* X_i *in* G *is denoted by* Π_{X_i}.

One reason why Bayesian networks are preferred for the analysis of gene expression data is because they can easily handle data with inconsistencies as well as imperfect models [Hartemink et al., 2001].

Bayesian networks do not however take into account the sequential order of the data. Additionally, they suffer from one main drawback which has to do with their directed acyclic graph (DAG) structure: they do not allow for loops in the network.

2.3.4 Dynamic Bayesian Networks

A *Dynamic Bayesian Network* (DBN) is an extension of the Bayesian network developed to consider the sequential nature of dynamic data. It is used when inferring regulatory relationships between nodes [Cho et al., 2007]. Unlike Bayesian networks, the Dynamic Bayesian Network works well with time series data and cyclic networks [Nakajima and Akutsu, 2014].

A DBN is described as a Bayesian network which has a time component to model time series data. In time series modeling, it is assumed that an event can only have an effect on another event in the future. For example, the gene expression in a network at time t can influence the gene expression at time $t+1$ but not the other way round. DBNs are particularly appropriate for representing stochastic temporal processes since each variable in a DBN is influenced by the previous one[Ghahramani, 2001]. As an example, consider the regulation of gene expression.

$$X_t := (x_1^t, ..., x_p^t)^T \in R^P$$

is a vector which represents the expression levels of p genes at time t.

A first-order Markov model

$$P(X^t|X^{t-1})$$

can be used to explain how the probabilistic distribution of gene expressions at time t is directly influenced only by those at time $t-1$

Based on the above hypothesis, the following equation can be used to denote the probability of gene expression levels over a time series of T steps, whereby the gene expression at time t given by X^t, is predicted based on the value of the gene expression at time $t-1$, given by X^{t-1}.

$$p(X^1, ..., X^t) = p(X^1)\Pi_{t=2}^T p(X^t|X^{t-1})$$

,

Since it is easier to interpret the semantics of DBNs, they are often preferred over the standard Bayesian networks. Normally, in a DBN, the direction of edges start from time $t-1$ and points to time t, and it is for this reason that DBNs are a natural choice

for representing gene regulatory relationships and other dynamic systems [Song et al., 2009].

These network models assume that the network topology stays the same over time, but in reality the real gene regulatory network in the cell changes its structure at different time points and when stimulated, for example, by the effects of environmental perturbations. Experiments are often of long duration during which the regulatory interactions between pairs of genes as well as the activities of the nodes may change and responses to stimuli may take varying amounts of time.

2.3.5 Existing approaches for modelling dynamic gene networks

Many approaches have recently been proposed, which use time-series gene expression data to build time-varying biological networks. To allow the network structure, inferred from the data, to vary between time segments, some methods have introduced change points in the time series. Lebre et al. [2010] introduced the autoregressive time-varying (ARTIVA) algorithm for the analysis of time-varying network topologies from time course data which has been generated from different processes. The interactions between genes are modelled using Reversible Jump Markov chain Monte Carlo (RJMCMC) and dynamic Bayesian networks. This is done for each segment of the time series.

A limitation of these approaches is that there is a prior assumption about the number of change points that can be observed; it cannot be automatically adjusted to suit the complexity of the observed data [Thorne and Stumpf, 2012].

The approach of Grzegorczyk et al. [2008] assigns each observation to a group, using an allocation sampler along with Bayesian Networks. The method allows the order of the observations to be interchangeable whereas the data is actually sequential. Werhli and Husmeier [2008] use a hierarchical modelling framework in which each individual dataset is used to infer a separate network structure. Unfortunately this approach only deals with steady state data using Bayesian networks and is not therefore applicable to time series data. Another approach using a hierarchical framework with several sources of time series data is that of Penfold et al. [2012]. Here it is used in conjunction with the non-parametric causal structure identification (CSI) algorithm [Penfold and Wild, 2011]. A single static network is built from the whole time series; the network structure is not allowed to change at different time points within the time series.

Thorne and Stumpf [2012] have proposed a methodology which uses the infinite Hidden Markov Model, also known as the *Hierachical Dirichlet Process Hidden Markov Model*, along with a Bayesian network representing the gene regulatory network at each hidden state. This model allows for the inference of gene regulatory networks which can change their structure at different time points whilst taking into consideration the sequential nature of data. A major goal of this project is to implement Thorne's method and extend it in a novel way by including the Dynamic Bayesian Network approach. This will accomodate cyclic regulatory relationships among the nodes in the gene regulatory network. A detailed explanation of this method is given in the next

chapter.

2.4 Some important concepts

In order to get a better understanding of the Hierarchical Dirichlet Process Hidden Markov Model method, it is important to first understand the concepts of Markov chains, Hidden Markov Model, Dirichlet Processes and Hierarchical Dirichlet Processes. A brief description of each of these concepts is given below.

2.4.1 Markov Chain

A *Markov chain* is a stochastic process depicting a sequence of possible events where the probability of the next event happening depends only on the current event's state. This process does not take into consideration the preceding sequence of events that occurred before the current one. Its state space is made up of a finite number of states and the probability of transitioning from state i to state j is denoted by P_{ij}.

For example, consider a sequence of random elements X_1, X_2,.... If the probability of X_{n+1} depends on X_n only, then the order of the sequence of elements is said to follow a Markov chain. A Markov chain is said to have stationary transition probabilities if the conditional distribution of X_{n+1} given X_n, does not depend on the value of n [Brooks et al., 2011]. In other words, in a stationary Markov process, the distribution of X_n is the same for all n.

The concept of a stationary distribution can also be defined as follows:

A (discrete-time) stochastic process X_n: $n \geq 0$ is stationary if for any time points $i_1,...,i_n$ and any $m \geq 0$, the joint distribution of $X_{i_1},...,X_{i_n}$ is the same as the joint distribution of $X_{i_1+m},...,X_{i_n+m}$ [sta, 2016]

The stationary aspect of a Markov process allows the proportion of time that a Markov chain spends in any particular state to be calculated. This is independent of the initial starting state. Let us consider the example taken from [sta, 2016], where we have a stationary process whereby for every n, $P(X_n=2)=\frac{1}{10}$. Thus, over a time span of 1000 time steps, we can assume that approximately 100 out of 1000 time steps will be in state 2 and over a time frame consisting of N time steps, around $\frac{N}{10}$ time steps will be in state 2. As $N \to \infty$, the amount of time the system will spend in state 2 will converge towards $\frac{1}{10}$.

Depending on the situation, it is possible to make a Markov chain stationary by ensuring that we have the correct initial distribution for X_0. If the Markov chain is in a stationary state, then the common distribution of all its states X_n is called the *stationary distribution of the Markov chain* [sta, 2016].

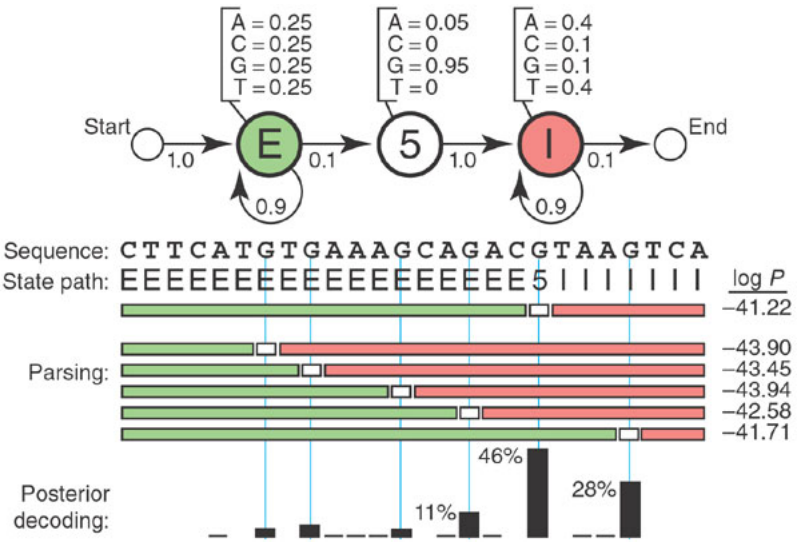
One way of extending the first order Markov model, such that the probability of a state is not restricted only to the probability of the previous state, is through the use of *Higher Order Markov chains*. This extension, which allows states to interact using higher order interactions, provides more flexibility than the more traditional Markov model. Consider the example of an nth order Markov model. In this model, the probability of variable X_i can depend on the previous variables $X_{i-1},...,X_{i-n}$, denoted by $P(X_i|X_{i-1},...,X_{i-n})$.

2.4.2 Hidden Markov Model

Hidden Markov model (HMM) is an extension the Markov model. This model, which is a type of dynamic Bayesian network, consists of two main components: a sequence of hidden states which follows a Markov process and a sequence of observations. It is assumed that the observations are dependent on the sequence of unobserved (hidden) states [Ghahramani, 2001].

Hidden Markov Models (HMMs) are the tool of choice when it comes to modelling time series data. A good definition which concisely explains the concept of a hidden Markov model is given in [Rabiner and Juang, 1986], which defines an HMM as *"a doubly stochastic process with an underlying stochastic process that is not observable (that is, it is hidden), but can only be observed through another set of stochastic processes that produce the sequence of observed symbols"* [Rabiner and Juang, 1986].

To get a better understanding of how the HMM functions, consider the following simple example of a HMM for a 5' splice site recognition, taken from [Eddy, 2004].



Source: Nature website
(http://www.nature.com/nbt/journal/v22/n10/full/nbt1004-1315.html)

Figure 2.4: A simple example of an HMM for the identification of a 5' splice site.

In the example, we have a DNA sequence containing an exon, a 5' splice site and an

intron. The aim is to identify the 5' splice site. Based on what we have so far, we can build an HMM consisting of three states where each of them corresponds to the exon, 5' splice site and intron respectively.

Each state in the HMM is associated with some emission and transition probabilities. Each hidden state emits some observed value based on the emission probabilities. In state *'I'*, symbol *'A'* is emitted with probability 0.4 and symbol *'C'* is emitted with probability 0.1. The transition probabilities associated with a state are the probabilities of moving from the current state to a new one. Therefore, starting from some initial state, a sequence of hidden states is formed based on the transition probabilities until the end state is reached. During that state-transition process, a sequence of observed symbols is emitted based on the emission probabilities at each state.

The main idea behind an Hidden Markov Model can be explained by the following two properties.

i The sequence of states is hidden. Only the values emitted by the states can be observed.

ii The hidden sequence of states follows a Markov chain whereby the value of the next state depends only on the current one. This ensures that the output of the states, which is the sequence of observed values, also satisfy a Markov property with regard to the states [Ghahramani, 2001].

2.4.3 Dirichlet Process

The *Dirichlet process*, an extension of the Dirichlet distribution, is a stochastic process whose domain comprises a set of probability distributions. Each draw taken from a Dirichlet process distribution is a distribution itself. Therefore, a Dirichlet Process can be described as a *distribution over distributions* [Teh, 2011]. Even though the distributions derived from a Dirichlet process are discrete, it is difficult to represent them by simply using a finite set of parameters . This is the reason why it is classified as a non-parametric model and, as a result, it is often used in Bayesian non-parametric models of data [Teh, 2011].

Non-parametric Bayesian models form part of a class of models whose parameters are not fixed in advance. Unlike traditional parametric models, which have a fixed and finite number of parameters that are normally predetermined, the parameters of non-parametric Bayesian models can be modified as needed to fit in with the data. As a result, the complexity of the inferred model can be adjusted according to the observed data.

Traditionally, a Dirichlet distribution is defined as:

$$P(x|\alpha) = \Pi_{i=1}^{M} x_i^{\alpha_i - 1}$$

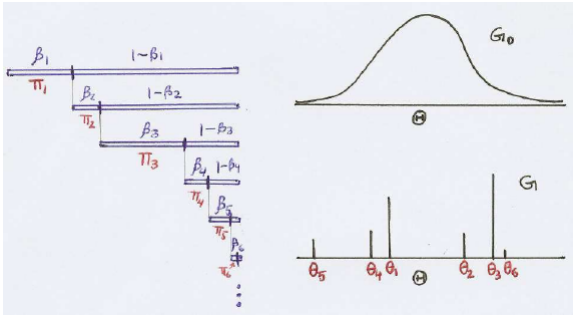
where x is a dimensional vector which takes parameters α_i for $i \in 1, ..., M$ and all $x_i > 0$. M represents the dimension of x such that $\sum_M x_i = 1$ [Thorne and Stumpf, 2012].

Since the observed values of the Dirichlet process are discrete and sum to one, it can be assumed that x_i define a discrete probability distribution over a set of outcomes $1,...,M$ while α_i is the number of observations of outcome i already seen.

"The Dirichlet process can thus be obtained as the limit of a symmetrical Dirichlet distribution with dimension M and concentration parameters $\frac{\alpha}{M}$ *as* $M \to \infty$*"*[Thorne and Stumpf, 2012].

One way of constructing the Dirichlet process was developed by *Sethuraman (1994)* and is known as "stick breaking". In this method, we assume that we have a stick of length 1. Let $\beta'_j \sim \text{Beta}(1,\gamma)$ for $j = 1,2,3,...$ and some concentration parameter γ. Consider β'_1, β'_2,... as fractions which we remove from the remainder of the stick every time. β_i can be derived by the lengths of the stick which we break each time.

$$\beta_i = \beta'_i \prod_{j=1}^{i-1}(1-\beta'_j)$$



Source: Zoubin Ghahramani Tutorial on Non-parametric Bayesian Methods (http://mlg.eng.cam.ac.uk/zoubin/talks/uai05tutorial-b.pdf)

Figure 2.5: Illustration of the stick breaking method

Therefore, a Dirichlet process, comprising of *concentration parameter* γ and *base measure* H which is denoted by $DP(\gamma,H)$ and $G \sim DP(\gamma,H)$, is represented by

$$G = \sum_{i=1}^{\infty} \beta_i \delta_{\theta_i}$$

where δ_θ represents an infinite sequence of discrete probability atoms taken from from the base measure [Thorne and Stumpf, 2012].

2.4.4 Hierarchical Dirichlet Process

The *Hierarchical Dirichlet process* (HDP) is an extension of the Dirichlet process and is normally applied when performing analysis between many different clusters of data. One important aspect of the HDP is that it can allow related groups of data to share clusters. It is composed of a Bayesian hierarchy *"where the base measure for a set*

of Dirichlet process is itself distributed according to a Dirichlet process" [Teh et al., 2012]. To construct a HDP, we only need to use a Dirichlet Process as the base measure of another Dirichlet Process [Thorne and Stumpf, 2012].

Based on the stick breaking method as described in [Teh et al., 2012], we get

$$G = \sum_{i=1}^{\infty} \pi_i \delta_{\theta_i}$$

where $\theta_i \sim H$, $\beta \sim GEM(\alpha)$ and $\pi \sim DP(\gamma, \beta)$

The flexibility that the Hierarchical Dirichlet Process, as a Bayesian non-parametric method, gives when applied to Hidden Markov Models makes it an attractive tool to use when analysing statistical data as it can easily adapt to fit in with the observed data [Thorne and Stumpf, 2012].

More details about the Hierarchical Dirichlet Process-Hidden Markov Model (HDP-HMM) is given in the *Algorithm* chapter.

The implemented HDP-HMM-DBN algorithm will first be evaluated using synthetic data to determine whether the predicted network structures are accurate. After which, it will be run on benchmark data available on the DREAM[1] (Dialogue for Reverse Engineering Assessment and Methods) challenge website, a site hosting challenges designed to assess the latest methods developed for gene network inference while providing better insight of systems biology.

Receiver Operating Characteristic (ROC) curves are used to test the benchmark evaluation. A ROC curve is a graphical plot useful for illustrating the performance of a binary classifier. By assessing the accuracy of the predictions made by the model, we can easily evaluate its performance and compare it to other models. The curve is created based on the true positive values and false positive values generated by the model when compared to some gold standard. By plotting the graph with the True Positive Rate (TPR) on the y-axis and the False Positive Rate (FPR) on the x-axis using different cutoff values, we get a comprehensive and visually appealing way of summarising the accuracy of the predictions. The main reason for using ROC curves is that they allow us to select the optimal models and discard the ones which are not deemed to be good enough. Matthews correlation coefficient gives us information about the performance at a specific cutoff.

More information about the DREAM challenge and the evaluation process is given in the *Analysis* chapter.

[1] DREAM Challenge: www.dreamchallenges.org

2.5 Cancer

An important task of this project is to analyse renal cancer data using the HDP-HMM-DBN algorithm.

Cancer is a major health concern around the world with tens of millions of people diagnosed every year; more than half of those diagnosed eventually succumb to it [Ma and Yu, 2006]. Due to progress made in cancer treatment in highly developed countries, there has been some decrease in cancer death rates, however this decrease is very small compared to estimates of increased cancer death rates over the next couple of years. It is predicted that the number of new cancer cases and cancer related deaths will increase to more than double globally over the next 20-40 years [Thun et al., 2010].

Cancer now ranks as the world's third most common cause of death with more than 14 million people newly diagnosed and 8.2 million people dying from it in 2012 [who, 2016]. By 2030, it is predicted that there will be approximately 26 million new cancer cases and 17 million deaths related to it each year [Thun et al., 2010]. This increase is largely driven by the fact that elderly people are more prone to cancer; population aging is a normal phenomenon around the world, especially in developed countries where people are living longer due to advances in medical science and improved lifestyle.

Biologically, cancer is the term used to refer to a group of related diseases which involves abnormal growth of cells in certain parts of the body. These cancerous cells can reproduce in an uncontrollable way and spread to surrounding healthy tissues, including organs and destroy them [nhs, 2016]. Cancer can happen anywhere in the human body and can affect any cell. A normal cell follows an orderly process whereby it grows and divides to form new cells and dies when it is old or damaged. When cells die, they are replaced by new ones. With cancer, the order of this process gets disturbed. Rather than dying, damaged cells continue to thrive and new cells are formed when not required. These cells accumulate and form growths known as tumors. Cancerous tumors are malignant, meaning that they can spread and affect surrounding tissues as well as other parts of the body which are not necessarily close to the location where the tumor developed [can, 2016b]. The process by which cancer cells spread to other areas of the body, not necessarily related to the part of the body where the cancer started, is known as *metastasis*. For example, cancer starting in the liver can end up affecting the kidneys.

A cancer tumor can have several mutations which make it unique: for instance if specimens from two breast cancer tumors are compared, the mutated genes contained in them will not be similar[can, 2016c]. [Hanahan and Weinberg, 2000] have reduced the complexity associated with cancer by explaining it through a series of six principles in their article *The Hallmarks of Cancer*. These principles explain how a normal cell changes to a cancer cell. In 2011, an update to the original article was published which proposed another four hallmarks [Hanahan and Weinberg, 2011]. A brief description of the original six hallmarks of cancer ensues.

1. *Self-sufficiency in growth signals*
   Growth factors are signalling molecules which control the activities of the cell.

For example they aid in: cellular proliferation and differentiation, survival, inflammation and tissue repair. Growth factors are an important requirement for normal cellular proliferation. Cancer cells do not require any stimulation from growth factors to proliferate; *"They are masters of their own destiny"*[Hanahan and Weinberg, 2011] in the sense that they produce their own growth signals and encourage their own propagation.

2. *Insensitivity to anti-growth signals*
   Cancer cells are insensitive to growth-preventing signals from surrounding cells. By avoiding and hindering those signals, cancer can continue on spreading.

3. *Evading apoptosis*
   Apoptosis is a programmed form of cell death which occurs as a normal process in the growth and development of an organism. In this form of cell death the collapse of the cell is characterised by: *"membrane blebbing, cell shrinkage, condensation of chromatin, and fragmentation of DNA followed by rapid engulfment of the corpse by neighbouring cells"* [Renehan et al., 2001]. In apoptosis, the cell is eliminated without any associated inflammation. Disruptions in apoptosis can lead to disorders such as autoimmune disorders, neurological diseases and cancer [Parsons and Green, 2010]. Cancer cells are resistant to death signals and evade apoptosis by avoiding the normal cell death cycle.

4. *Limitless replicative potential*
   Cancer cells spread and form tumors by avoiding apoptosis. They propagate as a result of abnormal and uncontrollable cell division and by being able to reproduce an infinite number of times. Cell division is an important part of normal cell growth and development. A normal cell divides a restricted number of times and this process is controlled by the telomere length. Telomeres are the repeated nucleotide sequences at the end of each DNA strand and help protect the chromosome. When a cell divides the telomeres shorten, and as division progresses the telomeres become shorter and shorter until eventually they become so short that the cell dies. Telomerase is the enzyme that maintains the telomeres and protects them from becoming too short during the course of cell division. Cancer cells can keep on replicating forever because they have the enzyme telomerase activated most of the time.

5. *Sustained angiogenesis*
   Angiogenesis is the process which involves the development of new blood vessels from pre-existing vasculature [ang, 2016]. It is a normal and important lifelong process during which embryonic tissues are formed. Capillaries are essential for carrying oxygen and nutrients from the bloodstream to the different tissues in the body. During normal development, once the necessary blood vessels are formed, the angiogenesis process stops. Cancer cells on the other hand always have the process of angiogenesis going on. The constant growth of new blood vessels causes proliferation of cancer cells by providing them with an adequate amount of oxygen and nutrients on which they can thrive. This process can cause a tumor to become malignant.

Angiogenesis is controlled by both activator and inhibitor molecules. Angiogenic inhibitors are used as a treatment for reducing cancer proliferation. [Nishida et al., 2006].

6. *Tissue invasion and metastasis*
   One alarming feature of cancer is its ability to spread to other parts of the body. Metastasis occurs when the cancer cells travel from the primary neoplasm to other parts of the body via the blood or lymph system, and affect distant tissue and organs. Most cancer related deaths occur because of metastases that are resistant to treatments. It is now known that a tumour cell metastasizes based on its interactions with homeostatic factors responsible for: tumour-cell growth, survival, angiogenesis, invasion and metastasis [Fidler, 2003].

To control their activity, cells make use of signalling pathways to transmit information within the cell. A signalling pathway refers to a group of molecules in a cell that interact with each other to enable the cell to function correctly. For example, those molecules can control functions such as cell division or cell death. When the first molecule in the pathway obtains a signal, it activates another molecule by conveying another signal. The second molecule in turn activates a third one and this process continues until the last molecule in the pathway receives a signal and is activated. As a result the cell function is performed. When abnormalities in the signalling pathway occur, for instance when signals are not correctly transmitted, this can lead to cancer. One way of inhibiting cancer growth and killing cancer cells is to block these pathways; drugs have been developed to do this [can, 2016a].

2.5.1 Renal Cell Carcinoma

Renal cell carcinoma (RCC) is a type of kidney cancer and is caused by a heterogeneous group of tumours that form in the tubules of the kidney [Bukowski and Novick, 2015]. Renal cell cancer is the most common type of kidney cancer in adults and more than 8 in every 10 kidney cancers diagnosed in the UK are this type[can, 2015]. At the time of diagnosis nearly one third of patients suffer from metastasis. This is due to the fact that RCC does not give any early signs or symptoms of it happening. Around 40% who undergo nephrectomy, (a procedure where part of or the entire kidney is removed), will ultimately develop this complication [Bukowski and Novick, 2015]. Its ability to spread without exhibiting any symptoms, and the fact that it is resistant to conventional chemotherapy, makes renal cell carcinoma a fearsome form of cancer. Studies have shown that cigarette smoking can double a person's risk of getting RCC and is a contributive factor to nearly one third of all cases [Motzer et al., 1996]. Obese people as well as those suffering from specific conditions, such as von Hippel-Lindau disease or hereditary papillary renal cell carcinoma, are also more at risk. Another risk factor is misuse of certain painkillers [rcc, 2016].

Clear cell carcinoma is one type of renal cell carcinoma, the other is papillary carcinoma: the classification being based on the structure and shape of the cancer cells. Renal clear cell carcinoma is the most common type of kidney cancer with approximately 92% of kidney cancer patients suffering from it [kid, 2016].

Currently no molecular method is available to predict how advanced RCC responds to targeted therapy. Existing clinical methods are limited and there is no way to identify patients who relapse after seemingly curative surgery [Galsky, 2013].

Renal cell cancers that have undergone metastasis display a large amount of tumour vascularity and Sunitinib is used to treat these abnormal growths. Sunitinib is a receptor tyrosine kinase inhibitor whose targets include: VEGFR, PDGFR and c-KIT. Sunitinib is often used to treat renal cancer [Vázquez et al., 2012]. Vascular Endothelial Growth Factor (VEGF) is an endothelial cell mitogen which controls the development of blood and lymphatic vessels as well as regulating homeostasis [veg, 2016]. VEGFR-1 and VEGFR-2 are closely-related receptor tyrosine kinases and are part of the VEGF family of receptors which are implicated in angiogenesis. VEGFR-1 regulates angiogenesis by the actions of: ligand-trapping, receptor homodimerization and heterodimerization. VEGFR-2 triggers a variety of signaling pathways [Rahimi, 2006]. Platelet-derived growth factor receptors (PDGF-R) are tyrosine kinase receptors which are important for embryonic and blood vessel development. These receptors play an important role in cell proliferation, survival, differentiation, chemotaxis and migration [pdg, 2016]. c-KIT, also known as CD117, is a tyrosine kinase receptor which binds to stem cell factor, a cytokine which plays an essential role in the development of blood cells. Modified forms of the c-KIT receptor can be found in certain types of cancer [cki, 2016].

Tyrosine kinase inhibitors have proven their efficacy when treating renal cell carcinoma (RCC) and other types of tumors such as gastrointestinal stromal tumors. Sunitinib is one type of tyrosine kinase inhibitor which has shown positive results in a study of *cytokine-refractory metastatic RCC patients* [Motzer et al., 2006].

To better understand the biology of renal cancer drug resistance and response, the HDP-HMM-DBN method is applied to existing renal cancer data sets which were obtained from cells exposed to the drug Sunitinib. This data is derived from Caki-1 cell line, *a human clear cell renal cell carcinoma (ccRCC) line that displays epithelial morphology and grows in adherent culture* [cak, 2016]. CAKI1 was selected as a representative of drug resistant cancer from analysis of a panel of 16 cell lines (Overton, personal communication).

In the longer term, these approaches would enable systems-wide dynamic modelling of renal cancer drug resistance mechanisms to enable in silico simulation of combination therapies towards more effective tools for cancer medicine.

Chapter 3

Algorithm

This chapter describes the algorithms in detail, in particular the Hierarchical Dirichlet Process - Hidden Markov Model - Dynamic Bayesian Network (HDP-HMM-DBN), implemented in this project, and the Metropolis Hastings Sampler algorithm which is used for the inference of gene regulatory network structures.

3.1 Hierarchical Dirichlet Process Hidden Markov Model

For the inference of time-varying gene regulatory networks from time series data, Thorne and Stumpf [2012] has proposed a methodology which uses the infinite Hidden Markov Model, also known as the *Hierachical Dirichlet Process Hidden Markov Model*, along with a Bayesian network representing the gene regulatory network corresponding to each hidden state. An important task of this project is to implement Thorne's method using a Dynamic Bayesian Network (DBN), to capture the interactions occurring between the genes at the time points corresponding to a particular hidden state. Use of the Dynamic Bayesian Network improves upon the method in [Thorne and Stumpf, 2012] in that it can deal with cyclic regulatory relationships among the nodes in the gene regulatory network.

This framework makes use of the Hierachical Dirichlet Process-Hidden Markov Model (HDP-HMM) to model the network structure at different time points while allowing it to vary it's topology [Thorne and Stumpf, 2012].

To model a hidden state sequence that changes over the course of time, the methodology of the *Infinite Hidden Markov model*, first introduced in [Beal et al., 2001] is used. This methodology describes how a standard hidden Markov model (HMM), consisting of a set of hidden states $s_1, \ldots s_n$,is enhanced such that the number of states is not limited to a specific number. The extended model can theoretically have an infinite number of potential states, although it is limited in practice.

Unlike a traditional HMM, where the number of states K is known in advance and the transitions between those states follow a Markov process, in an HDP-HMM the number of hidden states cannot be determined beforehand; they are generated based

on the available data, which ensures that the number of different states in the network structure can easily be modified to conform to the observed data.

Dirichlet processes are used to allow for an infinite number of transition parameters [Beal et al., 2001]. For example, if we consider the case of a traditional HMM, the probability of moving from state k to state l is given by the transition probability

$$\pi_{kl} = p(s_j = l | s_{j-1} = k)$$

That is, the next state l in the sequence is derived only from the current state k.

On the other hand in a HDP-HMM, a Dirichlet process prior is applied to the transition probabilities π_k of each state k in the hidden Markov model. In other words, each hidden state k has associated with it a Dirichlet process G_k and based on this, the next state is derived. A base measure G_0, common to all these Dirichlet processes, is shared amongst them such that $G_k \sim \mathrm{DP}(\alpha, G_0)$. This means that the distributions corresponding to the individual states are organised in a hierarchical structure, which allows groups of potential states to be shared. As a result, it facilitates transitions between them [Thorne and Stumpf, 2012].

The base measure is itself derived from a Dirichlet process $G_0 \sim DP(\gamma, H)$, where H is a prior over parameters for F_k which represent the emission distributions [Thorne and Stumpf, 2012].

Based on the concept of the stick breaking method of Sethuraman (1994), we have

$$G_0 = \sum_{l}^{\infty} \beta_l \delta_{\theta_l}$$

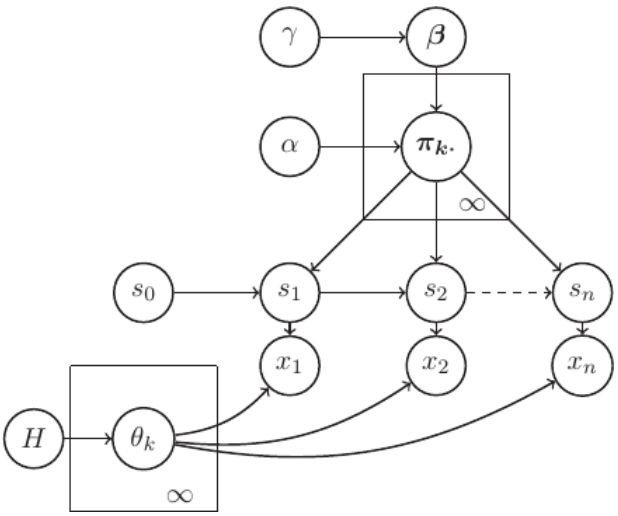
where θ_l is derived from H and with $\beta \sim GEM(\gamma)$.

Thus,

$$G_k = \sum_{l}^{\infty} \pi_{kl} \delta_{\theta_l}$$

with $\pi_k \sim DP(\alpha, \beta)$.

A model of the HDP-HMM is given in the figure below.



Source: [Thorne and Stumpf, 2012]

Figure 3.1: Graphical model of the HDP-HMM

3.1.1 Sticky HDP-HMM

For the method implemented in this thesis we use the sticky HDP-HMM, described in more detail in [Fox et al., 2008] and [Fox et al., 2011]. The sticky HDP-HMM makes use of an additional parameter which adds a bias to the transition probabilities between states. This allows the model to stay in its current state for a number of time steps, rather than changing at each step. In biological systems, such as gene regulatory networks, it is uncommon for the system to change state, or for the network topology to change, at each time step. This can be observed in gene expression datasets, where only a small number of transitions occur between different states across the time series [Thorne and Stumpf, 2012]. Use of sticky HDP-HMM is therefore more appropriate for the type of system being modelled.

3.1.2 Gibbs Sampling for the sticky HDP-HMM

When drawing samples from the hidden state sequence, a Gibbs sampling method is employed. This method updates each hidden state while taking into account the conditional probabilities of the hidden state s_i and the remaining hidden states s_{-i}.

In the original method developed in [Thorne and Stumpf, 2012], a standard Bayesian network methodology was used to model the gene regulatory network structure which corresponds to the hidden states of the HDP-HMM. However due to the restriction imposed by the DAG structure of a Bayesian network, it was not possible to derive all the potential set of network structures [Thorne and Stumpf, 2012]. For this project we use Dynamic Bayesian Networks to model the gene regulatory network structures. Each hidden state has associated with it a Dynamic Bayesian Network which describes the gene interactions taking place at a particular time point in the time series. Each time step corresponds to a hidden state.

After the sequence of hidden states has been derived, gene network structures can easily be sampled by the following steps:

i We first generate a potential set of parent nodes for a random gene.

ii Using the Metropolis Hastings acceptance probability value, we either accept the new set of parents or we reject it. If accepted, the network structure is modified to reflect that change.

3.1.3 Dynamic Bayesian Networks for representing the gene regulatory networks

Consider a data set consisting of p genes and n timepoints. Each gene i in the dataset is attributed a set of parents, denoted as *Parents(i)*. *Parents(i)* contains the genes whose expression level can have an effect the expression of gene i.

Assuming that the observations for the genes are indexed by $i \in 1...p$ and the time steps are indicated as $t \in 1...n$, we have:

$X_i^t = \sum_{j \in Parents(i)} a_j X_j^{t-1} + \varepsilon$, where ε represents noise and is denoted by $\varepsilon \sim N(0, \sigma^2)$.

The gene regulatory network is a network with directed edges from each of gene i parents to gene i, such that $j \rightarrow i$ for each j in $Parents(i)$.

The HDP-HMM-DBN method is summarised in Algorithm 1.

3.1.4 HDP-HMM-DBN algorithm

Set α, κ to fixed numbers (1.0)
Create an array β with 2 elements set to 1.0
Initialise state sequence s from 1 to $n-1$ (set all to 1)
Define a burn in value
Initialise 2 lists (finalList,finalSeq) to store the results of the final list of networks and state sequence
Generate random network
Create list of networks *netlist* and append random network(initial list of just one network)
Set K to the current number of networks (1)
for *j in 1 to 10000* **do**
  for *k in 1 to K* **do**
    Update network structure $netlist[k]$ based on X at timepoints where $s[t] = k$ (metropolis hastings update)
  end
  for *t in 1 to n* **do**
    Generate a random network for $K+1$
    for *k in 1 to K* **do**
      Calculate $p(s[t] = k)$ (the state probability)
    end
    Calculate StateProb for K+1
    Normalise $p(s[t] = k)$ so that they sum to 1
    Generate a random integer l from 1 to $K+1$ with probability $p(s[t] = k)$
    Set $s[t] = l$
    if $l = K+1$ **then**
      Set $K = K+1$, add network to list of networks
      Add $\beta[K+1]$, set to 1.0
    end
    for *k in 1 to K* **do**
      if *there are no s[t]=k* **then**
        Delete $netlist[k]$
        Delete $\beta[K]$
        Set all $s[t] > k$ to $s[t]-1$
        Set $K = K-1$
      end
    end
    if $j >$ *burn-in value* **then**
      store s in finalSeq list store netList contents to finalList
    end
  end
end
Create adjacency matrix based on finalList (contains probability for each edge)

Algorithm 1: HDP-HMM DBN algorithm

The *burn-in* value in the algorithm represents the initial number of iterations which are disregarded before we start collecting samples. This is done to ensure that only meaningful samples are collected, by minimising the effect that initial values have on the posterior inference.

To deal with the computational complexity associated with the inference of the Dynamic Bayesian Networks, a limit is placed upon the number of parents associated with a node. The value given to the number of potential parents depends on the size of the data and the number of replicates it contains. Replicates in gene expression data refer to repeated measurements which are performed in microarray experiments

to minimise the amount of noise associated with the experimental data. This helps reduce ambiguity and variability in the results.

The next section looks into the Metropolis Hastings algorithm, which is used in our method to update the network structure associated with each hidden state as it evolves over time.

3.2 Metropolis-Hastings Algorithm

Metropolis-Hastings (MH) algorithm is a powerful Markov Chain Monte Carlo method which makes use of a Markov chain when drawing random samples from a probability distribution. One important aspect of the MH algorithm is that it can be used to simulate multivariate distributions, especially those which have a high number of dimensions. In the HDP-HMM-DBN method, the Metropolis Hastings Sampler is used for the inference of Dynamic Bayesian Network structures corresponding to each hidden state of the HDP-HMM by drawing samples to represent the structure of the network. We begin with an initial set of nodes and then, over a number of iterations, we propose a potential parents set for each node which we either accept or reject, depending upon the value of the Metropolis Hastings acceptance probability.

The two algorithms below give a brief description of the MH sampler as used in our method.

```
Initialise parents of all genes
for n in 1 to 10000 do
    Select a random gene i
    Generate potential new parent set (do not overwrite current one)
    Calculate Metropolis-Hastings acceptance probability
    Generate uniform random number 0 < r < 1
    if r < acceptance prob then
        set parents of gene i to new set
    end
    if n > burn in and n modulo 10 = 0 then
        store parents of all genes in list
    end
end
```

Algorithm 2: Metropolis Hastings algorithm

```
Generate uniform random number 0 < r < 1
if r < addprob then
    add a new parent (choose a gene uniformly at random from those not currently parents)
    calculate q(Parents(i) → Parents(i)' and q(Parents(i)' → Parents(i))
end
else
    delete a parent (choose a parent uniformly at random and delete)
    calculate q(Parents(i) → Parents(i)' and q(Parents(i)' → Parents(i))
end
```

Algorithm 3: Algorithm to generate a proposal of a new parent set for random gene i

The Metropolis Hastings algorithm in itself is composed of three main components.

(i) Generating a proposal sample.

(ii) Calculating the acceptance probability.

(iii) Accepting or rejecting the candidate sample based on the acceptance probability.

The first step of the algorithm consists of assigning a sample value to a variable. In our case, we select a node, representing a gene, uniformly at random. We then find the parents of that node by finding nodes where there exists a directed edge to the selected child node. Next, we generate a potential new parent set for that child node.

Proposal Distribution: The proposal distribution is the conditional probability of proposing a new state x' given x. In this step, we generate proposal of a new parent set. We first generate a uniform random number between 0 and 1 and assign it to a variable, for e.g *r*.

If the value of *r* is less than the *addprob* value and the number of parents of that child node is below a certain threshold, we select a node which is not already a parent of the child node and add it as a potential parent node. The network is then updated by adding an edge from the potential parent node to the child node. We next calculate the probabilities of proposing to move:

- from the current parent set to the potential parent set, represented as $q(Parents(i) \rightarrow Parents(i)')$.
- from the potential parent set to the current one, $q(Parents(i)' \rightarrow Parents(i))$.

Now, if the value of *r* is above the *addprob* value and/or the number of parents of the child node is above the threshold value, we randomly select a node which is currently a parent of the child node and remove the edge between it and the child node. After the network has been updated, we calculate the probabilities of proposing to move.

Acceptance Function: In this function, the acceptance probability is calculated and based on that, we decide to either accept or reject the candidate parent set. The MH algorithm consists of a Markov process which is designed to satisfy the following two constraints:

1. The sampler should aim to visit higher-density regions and return the majority of the samples from these regions.
2. The sampler should explore the sample space by randomly moving about and ensuring that it does not get stuck in the same site.

It is important that the MH acceptance function satisfies the above conditions because this ensures that the stationary distribution of values produced by the MH algorithm is closer to the target distribution that we are interested in.

The *Metropolis-Hastings* acceptance probability of a proposed new set of parents $Parents(i)'$ is calculated using the following equation:

$$a = \frac{p(Parents(i)'|X)q(Parents(i)' \rightarrow Parents(i))}{p(Parents(i)|X)q(Parents(i) \rightarrow Parents(i)')}$$

where $q(Parents(i) \rightarrow Parents(i)')$ is the probability of proposing to move from parent set $Parents(i)$ to $Parents(i)'$.

Accept/Reject a proposal: After the acceptance probability has been calculated, we use it to either accept or reject the proposal. In practice, we generate a random number uniformly between 0 and 1. If this value is smaller than the acceptance probability, we accept the proposal by replacing the existing parents set of the child node with the potential set of parent nodes derived in the proposal distribution step. Otherwise we reject the proposal and keep the current parents set, in which case no update is made to the network.

In the next chapter, we talk about how the HDP-HMM-DBN method is implemented using the R language and give code extracts of the main functions.

Chapter 4

Implementation

4.1 R language

R, an open-source software project available under the GNU general public license, is the language of choice when it comes to computing and graphics in the statistical field. It is often used by statisticians and scientists for performing tasks such as: time-series data analysis, classification, predictive modelling and data visualisation. Its core strengths are: the large variety of mathematical functions available for manipulating and analysing data, the ease with which user-written functions and scripts can be incorporated using the object-oriented paradigm, excellent facilities for creating graphical plots.[rla, 2016].

Key R features that are used in the HDP-HMM-BDN algorithm implementation:

- matrix,list,vectors are used to store the data/values
- mathematical functions such as exponentiate, logarithm, solve, transpose are used for data manipulation
- graph plotting functions

R is highly extensible and quite easy to integrate with other applications, facilitating the use of other packages.

The two R packages that are the main imports in our algorithm implementation are *Igraph* and *rTMA*. A brief description of each follows:

4.1.1 Igraph package

Igraph [igr, 2016] is a set of tools geared towards network analysis. It is mostly used for the creation, manipulation and visualisation of graphs and networks. Bindings for igraph are available in R, Python and C/C^{++}. The one that we are using for our project is the *R/igraph* package, an *R* package of the Igraph network analysis library. All

networks in the HDP-HMM-DBN method are created using the Igraph function and need to be manipulated using specific functions.

4.1.2 rTMA package

rTMA [rtm, 2016] is an R package, developed by the Overton group, for the analysis of tissue microarray (TMA) data. Tissue microarrays (TMAs) are composite paraffin blocks on which tissue cores are placed in an array in order to perform histological analysis. Histology is the study of tissues and cells using a microscope and is an important tool used to monitor the progress of treatments, for example by monitoring how cancer cells react to certain drugs [his, 2016]. The tissue samples come from many different sources and a thousand or more may be placed together on a single histologic slide. This allows a large variety of specimens to be analysed simultaneously in similar and standardised conditions [Jawhar, 2009]. Tissue microarray is an efficient tool, featuring high throughput molecular analysis of tissues, which aids the discovery of new diagnostic and prognostic markers and targets in human cancer [Jawhar, 2009].

The rTMA package has been implemented to satisfy the requirement for a simple and effective tool with which to analyse TMAs. rTMA takes as input a csv/tsv file containing TMA data in the form of quantitative protein expression. The data is stored in a TMA object, which also stores associated clinical data [Lubbock, 2016]. The ComBat algorithm [Johnson et al., 2007] can be applied when several TMA slides are used. The ComBat algorithm makes use of an empirical Bayes method to minimise experimental variations of a biological or technical nature, which occur between the slides. The rTMA package allows for a variety of analyses to be performed on the TMA data. These include protein marker expression visualisation, correlation and relevance networks analysis [Lubbock, 2016]. The rTMA functionality which we use mostly in our HDP-HMM-DBN algorithm is a network inference method called *GABI*.

GABI is a novel algorithm which has been implemented by the Overton group and is aimed towards performing relevance thresholding in networks. It is designed for the inference of small-scale networks using TMA data. GABI has been implemented to deal with issues which Spearman correlation and Mutual information cannot properly manage. For example, although quite resilient to noise, Spearman correlation can miss certain classes of protein-protein interaction, and Mutual information can overload the user by identifying all types of statistical correlations in the network. Additionally, the data contained in TMAs is often highly related, and this can make a standard correlation network produced from that data difficult to interpret. In GABI, Spearman correlation is used to check for pairings between candidate proteins as part of the protein-protein interactions. Symmetric uncertainty, if required, is applied to the output of the Spearman correlation to determine whether the edges are signed or not. The aim is to reduce the number of hypothesis tests that need to be done [Lubbock, 2016]. A relevance thresholding procedure is then applied to get rid of edges which are below a certain relevance threshold value. The final output of GABI is a network that contains only "high confidence" relationships, which can be either signed or unsigned depending on whether Spearman correlation was applied or not.

4.2 Code Extract

In this section, we provide an overview of the method implementation. Extracts of code, from a trimmed down version of the important parts of the implementation, are given along with explanations.

4.2.1 Code Extract: Network class

The HDP-HMM-DBN implementation requires the use of graph data structures to represent the genes and the interactions occurring between them. During the initial development phase of the method, a custom graph class was implemented. This *Network* class had the basic functionality required for graph manipulation, and in developing it a good understanding of the working of graphs and their associated operations was obtained. Eventually we moved to the complete igraph package, which offers much greater functionality and visualisation capability. Our prototype Network class was simply replaced with the Igraph's class since the rTMA package also makes use of Igraph, and this allowed for a seamless integration with the HDP-HMM-DBN algorithm.

The Network class implements the required functions to manipulate graphs during the simulation. Some of the functionalities include: add/delete nodes and edges to the network, search for the children/parents of a node and search for edges.

4.2.2 Code Extract: Metropolis Hastings Algorithm

Following is an extract of the main function from the Metropolis Hastings Sampler algorithm. It shows the implementation of the main iteration, including inline comments for the variables and blocks of expression.

```
# Input: gene expression matrix and a network
# Output: updated network

# constants
addprob <- 0.5
delprob <- (1 - addprob)
burnIn <- 10
numIterations <- 1000

# Main MH-sampler function

mhsampler_new <- function(gexprX, gexprY, gnet, priornw)
{
  geneExpression <- gexprX
  geneNetwork <- gnet

  # number of genes in the gene expression matrix
  p <- ncol(geneExpression)

  # max. num of edges of network with p nodes = (p-1)p
  e <- 2 * p

```

```
23    # timesteps/observations in the gene expression matrix
24    t <- nrow(geneExpression)
25
26    # main iteration
27    for (i in 1:numIterations) {
28      # choose gene uniformly at random
29      randomGene <- sample.int(p, 1)
30
31      # find chosen gene parents
32      randomGeneParents <- findParents(geneNetwork, randomGene)
33
34      # generate potential new parentset
35      potentialParentSet <- genPotentialParentSet(geneNetwork,
            randomGene)
36
37      randomGeneParents_updated <-
38        findParents(potentialParentSet$netwk, randomGene)
39
40      # calculate Metropolis Hastings acceptance prob
41      X <- gexprX[, unlist(randomGeneParents), drop = FALSE]
42      y <- gexprY[, randomGene, drop = FALSE]
43      acceptanceProb_old <- genAcceptanceProbability(...)
44      acceptanceProb_updated <- genAcceptanceProbability(...)
45
46      acceptanceProb <- exp(acceptanceProb_updated - acceptanceProb_
            old)
47                            * (potentialParentSet$probTo /
                                 potentialParentSet$probFrom)
48
49      #generate uniform random number between 0 and 1
50      r <- runif(1, 0, 1.0)
51
52      if (r < acceptanceProb) {
53        # set parents of gene to new set
54        geneNetwork <- potentialParentSet$netwk
55      }
56    }
57    return(geneNetwork)
58  }
```

Listing 4.1: Metropolis-Hastings Sampler Algorithm

The main function in the *Metropolis-Hastings* algorithm takes four parameters as input: two blocks of rows at two consecutive time points from the gene expression matrix, the gene network to be updated and an optional prior network which in our case is the GABI network. In the main iteration, a node representing gene i is selected uniformly at random. We next find the set of parents for that gene, denoted by *Parents(i)*. Thereafter a proposal for a new parent set for gene i is generated.

To derive the structure of the network, we initially only need to know what the set of parents of each gene is. Using a Bayesian approach, we sample from the posterior distribution of the parent sets for each gene such that:

$$p(Parents(i)|X) \propto \int p(X|Parents(i),a)p(Parents(i))p(a)da$$

where X is the gene expression matrix and i is the random gene selected initially.

We can use the *Metropolis-Hastings* sampler to provide us with samples from $p(Parents(i)|X)$ and *Zellner's g-prior*[Zellner, 1986] to calculate $\int p(X|Parents(i),a)p(Parents(i))p(a)da$. Zellner's g-prior is used because it gives a simple equation for the integral over the a (that is not really important in our case). The prior on parents $p(Parents(i))$ is either uniform, or based on GABI input.

We next require a proposal distribution which can be used to derive new sets of parents. To transition from the current set of parents to a new one, there are two possible types of move that can be made. Either we add a new parent to the exiting parent set of gene i, or delete an existing one from that set. As part of the transitioning process, a uniform random number r is generated between 0 and 1 and based on this, gene i parent set is either updated or left unchanged.

In the initial implementation phase of the *Metropolis Hastings* sampler algorithm, there is an initial burn in set of iterations which are discarded. The network is then stored in a list, and updated after every subsequent tenth iteration. When the sampler algorithm was integrated with the method's main algorithm, the code was modified as follows: after running the main loop a pre-defined number of times, only one (updated) network is returned, and this is used in the *HDP_HMM_DBN* algorithm.

The *Metropolis Hastings* sampler algorithm is also composed of two other functions, namely the *genPotentialParentSet* function and the *genAcceptanceProbabilty* functions.

The *genPotentialParentSet* function takes as input a network and gene i, which is the random gene chosen in the *MH-Sampler* main function. It returns as output an updated version of the network and the probabilities of proposing to move: $q(Parents(i) \rightarrow Parents(i)')$ and $q(Parents(i)' \rightarrow Parents(i))$

```
# generate proposal of a new parent set
genPotentialParentSet <- function(network, childnodeId)
{...}
```

Listing 4.2: Function to generate proposal of a new parent set for gene i

The *genAcceptanceProbabilty* function is used to generate the *Metropolis Hastings* acceptance probability, which is in turn used to decide whether to update the parents set of gene i to a new one. This function takes as input the network, an optional prior (GABI) network , gene expression matrix, columns in the gene expression matrix corresponding to genes which are parents of gene i (X), columns in the gene expression matrix corresponding to gene i (y), gene i and gene i's parents.

A set of mathematical calculations is performed, to calculate the acceptance probability value. Further calculations are performed if a prior network is given. If a GABI network is provided as prior, we test each edge from $Parents(i)$ to gene i by verifying if it is in the prior network; the final output probability is adjusted depending on whether the edge is in the prior or not. The output of the *genAcceptanceProbabilty* function is an acceptance probability value, used in the main *Metropolis Hastings* sampler algorithm when calculating the probability of a new set of parents for gene i.

```
# generate MH acceptance probability - calculates the probability of
    a set of parents
```

```
genAcceptanceProbability <- function(nw, priornw, geneExpr, X, y, gene,
    geneParents)
{...}
```

Listing 4.3: Function to generate proposal of a new parent set for gene i

4.2.3 Code Extract: HDP-HMM-DBN Algorithm

The *HDP-HMM-DBN* algorithm takes as input several parameters which are listed and described below:

- *inputFile*: The file contains the time series data which eventually gets converted to a gene regulatory network adjacency matrix.
- *blocksize*: This parameter is the number of consecutive rows in a block of gene expression data and represents the number of replicates in the experiment from which the data is derived.
- *sanitize*: The *sanitize* parameter takes as value either TRUE or FALSE and if assigned to TRUE, it converts 'NAs' in the data to zero.
- *gabi*: This argument also takes a Boolean value which when set to TRUE creates a directed GABI network from the time series data. This directed network is then used as prior in the *HDP-HMM-DBN* algorithm.
- *J*: *J* is the number of iterations in the main loop of the *HDP-HMM-DBN* algorithm. It has a default value of 100 which get overwritten by the value input by the user.
- *burnInValue*: This argument represents the initial number of iterations which are to be ignored before we start collecting gene regulatory networks that evolve over time.

The first thing the HDP-HMM-DBN algorithm does is to check if GABI is set to TRUE. If it is, the time series data provided as input is used to generate a directed GABI network which is then used as prior. This network is generated by running the GABI algorithm provided in the rTMA package. Several variables are initialised for use in the algorithm. One of them is a vector s ($s \leftarrow rep(1,n)$), which stores the hidden states sequence.

Before entering the main loop, we generate a random network which gets stored as the initial network in a list. Inside the main function's first inner loop, we update all of the network structures in *netList* based on the gene expression data at specific time points using the *Metropolis-Hastings* algorithm. In the second inner loop, we generate another random network to potentially add to our list of networks. We then choose the network from the list and this new network, that best fits each timepoint in the timeseries. This step is iterated a number of times until we have a list of networks depicting their temporal evolution over the time steps corresponding to the time series input file. We only keep the networks which are generated after the initial burn in stage to ensure that we get only "good" samples. Upon completion of the main loop we

create an adjacency matrix, which stores the probability of each edge in the network. The adjaceny matrix is written to a file for subsequent use in the benchmarking step.

```
# Input: gene expression matrix and optional parameters
# Output: list of networks at different time points showing the
    evolution of gene regulatory networks over time

# load rTMA and other packages
library(rTMA)
library(igraph)
library(parallel)

# sanitize data to remove nulls/na
sanitizeData <- function(X) {...}

# e.g. hdp_dbn(INPUT, blocksize=5,J=10,burnInVal=5)
hdp_dbn <- function(inputFile,blocksize,sanitize=FALSE,gabi=FALSE,J
    =100,burnInVal=80)
{
  # read test or GABI data (csv format)
  X <- read.csv(inputFile, header = TRUE)

  if (sanitize==TRUE) X <- sanitizeData(X)

  #define gene expression matrix
  geneXprMatrix <- scale(X)

  if(gabi==TRUE) {
    # create TMA object using input data
    tmaObj <- tma(X)
    ...
    # create directed Gabi network
    ...
  }

  # num of nodes in network
  p <- ncol(geneXprMatrix)
  # num of edges = 2 x p
  e <- 2 * p

  # define constants to be used in the method

  # create an array betaArr with 2 elements set to 1.0
  betaArr <- c(1.0, 1.0)
  # number of rows in a block - replicates in expression matrix
  bsize <<- blocksize
  # num of observations/timesteps
  n <- nrow(geneXprMatrix)/bsize-1
  #initialise state sequence s from 1 to n, set all to 1
  s <- rep(1,n)

  #generate random network
  randomNetwork <- createIgraphNetwork(p,e)

  # store randomNetwork in list
  netList <- list(randomNetwork)

  # set K to current number of networks in list
  K <- length(netList)

  # outer loop
  for (j in 1:J) {

    #First inner loop
```

```
60      for(k in 1:K) {
61        #update network structure netlist[k] based on gene expression
             X at timepoints where s[t] = k (metropolis hastings update)
62        netw <- netList[[k]]
63
64        # update netw structure based on gexp_timepoint using
             metropolis hastings update
65        updated_netw <- mhsampler_new(...)
66
67        #set netList[k] to be that particular network
68        netList[[k]]<-updated_netw
69      }
70
71      #Second inner loop
72      for (t in 1:n) {
73        #generate a random network for K+1
74        newNet <- createIgraphNetwork(p,e)
75
76        #initialise stateProb to be a vector/list of length k+1
77        stateProb <- vector(mode="numeric", length = K+1)
78
79        for (k2 in 1:K) {
80          #calculate state probability for k2
81          stateProb[k2] <- calculateStateProb(...)
82        }
83
84        # calculate stateProb for K+1
85        stateProb[K+1] <- calculateStateProb(...)
86
87        # Normalise p(s[t] = k) so that they sum to 1
88        stateProbNor <- ...
89
90        # generate random integer l from 1 to K+1 with prob p(s[t] = k
             )
91        l <- sample(1:(K+1),1, prob=stateProbNor)
92
93        # set s[t] = l
94        s[t] <- l
95        if(l == K+1){
96          K <- K+1
97          #add network to list of networks
98          netList[[K]] <- newNet
99        }
100
101       for (k in K:1) { ... }
102     }
103
104   # store netList contents in finalList after burnIn
105     if (j > burnInVal)
106     { ... }
107
108   }#end outer loop
109
110   #Calculate a probability for each edge and store in an adjacency
        matrix as final output
111
112 }
```

Listing 4.4: HDP-HMM-DBN Algorithm

Other functions which form part of the *HDP-HMM-DBN* algorithm are listed below, along with a brief description of their functionality.

The *getblock* function takes as input the gene expression matrix and a block number. It returns a set of consecutive rows from the matrix which corresponds to the block number given as argument. The number of rows contained in one block corresponds to the number of replicates in the data.

```
# Calculate index positions for gene expression data at timepoints
    where s[t] = k and return the specified block number from given
    matrix M
getblock <- function (M, bnum)
{ ... }
```

Listing 4.5: Function to calculate a set of consecutive rows in the gene expression matrix corresponding to a block number

The *getblocks* function has as parameters the gene expression matrix and a list of block numbers. It returns a set of rows from the gene expression matrix, which contains the blocks of data given as second parameter.

```
# returns a matrix containg the blocks given as argument
getblocks <- function(M, bnumlist)
{...}
```

Listing 4.6: Function to return a set of rows in the gene expression matrix corresponding to a list of block numbers

The *calculateStateProb* function takes as input several parameters, including: the state sequence, the network and the GABI network, and uses them to calculate the probability of transitioning to a different state. Several steps are taken to determine the transition probability:

1. First, a new state sequence is derived, which excludes the element at position *j* in the original sequence.
2. A transition matrix is created, based on the new state sequence.
3. Next, we calculate the number of observed transitions from state *j-1* to *k2* and from state *k2* to *j+1*
4. Finally, we derive a set of rows from the gene expression matrix such that the row numbers correspond to the indices of the state sequence and its value equals to *k2*. We then derive another set of rows based on the state sequence indices incremented by one. Using this, we calculate the likelihood of the data given the network structure currently assigned to each state using the *genAcceptanceProbabilityAllNodes* function.

```
# Calculate state probability p(s[t] = K)
calculateStateProb <- function(...)
{...}
```

Listing 4.7: Function to calculate the state probability of transitioning to a different state

The *genAcceptanceProbabilityAllNodes* function uses the *genAcceptanceProbability* function to calculate the acceptance probability for each node in the network, and for each of their parents and return their sum.

```
# Calculate Acceptance probability for for all nodes in the network
    network[k] along with their parents
genAcceptanceProbabilityAllNodes <- function(...)
{...}
```

Listing 4.8: Function to return the sum of the acceptance probability for all nodes in the network

The logical step that follows is to conduct experiments based on the method described and implemented in this chapter. The next chapter describes the benchmarking and analysis process.

Chapter 5

Benchmarking and Analysis

This chapter evaluates the algorithm presented in the thesis through benchmarking and analysis of the results obtained by running our implementation of the algorithm.

The HDP-HMM-DBN algorithm is evaluated on three data sets:

1. Synthetic/test data
2. DREAM Challenge data
3. Renal Cancer data

In the course of implementing the HDP-HMM-DBN algorithm, the output and performance of the code were tested repeatedly with synthetic data (first data set in the list above) generated from the DBN model and using a known network. This was a useful step to debug the code and as an initial validation of the algorithm's correctness and behaviour. The tests were carried out on a small scale: the test data comprising 10 genes, identified by $A, B, ..., J$, with 20 observations in 3 sets of replicates. After the initial set of tests, the DREAM4 insilico time series datasets are used for evaluation and benchmarking. The algorithm is then applied to renal cancer drug response time course data obtained from the Overton[1] group, in collaboration with Grant Stewart[2] (Consultant Urological Surgeon, Western General Hospital).

Data Input Format

The time series data used as input to our algorithm needs to be in a specific format: each column represents a gene expression profile, and each row represents a microarray experiment at a given time step. Replicates occur when we have multiple rows with the same time value. Replicate measurements are done to increase the confidence level of the results. The number of observations corresponds to the number of rows in one set of replicates.

An example of the input data format is given below for 9 genes and 3 replicates. The number of observations in this example is 3 as there are 3 rows for each time step.

[1] Overton Group: http://www.hgu.mrc.ac.uk/people/i.overton.html

[2] Stewart Group: http://www.ed.ac.uk/surgery/staff/surgical-profiles/grant-stewart

```
Time G1 G2 G3 G4 G5 G6 G7 G8 G9
10   .  .  .  .  .  .  .  .  .
20   .  .  .  .  .  .  .  .  .
30   .  .  .  .  .  .  .  .  .
10   .  .  .  .  .  .  .  .  .
20   .  .  .  .  .  .  .  .  .
30   .  .  .  .  .  .  .  .  .
10   .  .  .  .  .  .  .  .  .
20   .  .  .  .  .  .  .  .  .
30   .  .  .  .  .  .  .  .  .
```

Listing 5.1: Example of input data format

5.1 DREAM Challenge Data

DREAM challenge[3] is a non-profit, open science effort, designed and maintained by a group of researchers from different areas of specialisation. The purpose is to gain insight into the fundamentals of systems biology and translational medicine, while facilitating improvements in these biological sciences by giving researchers access to novel data. The challenges enable participants to propose solutions to problems, or to test their solutions or methodologies against the gold standard to see how well they fare. New and better computational models are thus developed and shared by the scientific community. These models can be used to make significant discoveries and solve complex problems. The knowledge gained while taking part in a challenge is stored on Synapse, a software platform which allows scientists to share their research.

5.1.1 DREAM4 Insilico Networks

The insilico network challenge provides participants with simulated steady-state and time-series data, which can be used to reverse engineer gene regulatory networks. Using the given insilico gene expression datasets, participants should derive the network structure and optionally predict how the networks will react to a set of perturbations (not included in the challenge) [Greenfield et al., 2010]. The challenge is composed of three sub-challenges and each sub-challenge provides data for 5 different networks. Each sub-challenge tests how consistently the method under investigation predicts the topology of the gold standard networks.

For the benchmarking, we are using the time series data of the *Insilico_size10* and *Insilico_size100* sub-challenges to predict the network structure given the provided datasets.

Insilico_Size10 sub-challenge This sub-challenge provides datasets of type: wild-type, knockouts, knockdowns, multifactorial perturbations and time-series, however only the time-series data is relevant for this evaluation. The data consists

[3]DREAM Challenge: www.dreamchallenges.org

of networks of 10 nodes and the goal is to predict the directed unsigned edges found in these networks.

Insilico_Size100 sub-challenge The Insilico size 100 sub-challenge is similar to Insilico size 10, except that the networks are of size 100 instead of 10 and multifactorial perturbation datasets are not provided, however, we will only use time-series data as with the previous sub-challenge.

The HDP-HMM-DBN algorithm is benchmarked using time series data from both the above sub-challenges. For each sub-challenge, 5 files corresponding to 5 different networks are given. Each file contains time series data depicting how the network reacts when a perturbation is added and removed. For insilico size 10, 5 replicate data sets are provided, while for insilico size 100, 10 are provided. Each data set contains 21 time points; the first half (from t=0 to t=500, interval=50) shows how the network responds to a perturbation added at t=0, the remaining time steps depict how the network relaxes when the perturbation is removed at t=500.

Machine learning prediction methods are increasingly being used in the field of Bioinformatics. Evaluating the performance of these methods is an important step which needs to be carried out before they are used on real-world data. Similarly, the HDP-HMM-DBN algorithm is first parameterised and benchmarked using DREAM4 datasets and is then applied to real-world renal cancer data.

5.1.1.1 Evaluation Steps

The steps followed for the evaluation of our method using the benchmarking data are listed below.

Step 1 Run the algorithm against DREAM4 data

Step 2 Compare output with the gold standard networks

Step 3 Perform cross-validation

Step 4 Find area under the ROC curve

Step 5 Derive meta-analysis graphs

5.1.1.2 Methods and Metrics of Evaluation

We first list and briefly explain some of the methods and metrics (and what measures they provide) used for evaluation and discussion that follows.

1. *Receiver Operating Characteristics (ROC) curves*

   There are several performance evaluation measures which, when used in conjunction with the ROC curve (introduced in Section 2.4.4), provide an intuitive way of visualising the prediction performance. There are multiple tools available to display and analyse ROC curves, for instance: ROCR package, easyROC

web tool, and pROC among others. For our purposes we opted for the *ROCR*[4] package because of its ease of use and because it integrates well with R's built-in graphics functions.

2. *Matthews Correlation Coefficient (MCC)*

   MCC provides a measure of the quality of the binary classifier method being tested, by taking into account true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN). True positives are defined as the correctly predicted edges, false positives represent the predicted edges which do not exist in the gold standard network, true negatives are the correctly identified non-edges while false negatives are the actual edges in the gold standard identified as non-edges by the algorithm [Lund, 2005].

   The equation to calculate MCC is given by:

$$\frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

   The values returned by the MCC measure range between -1 and +1, where +1 represents an exact prediction made by the method, 0 indicates the result of a random prediction and values less than 0 are worse than random, with -1 being the inverse of a perfect prediction.

3. *Cross-validation*

   Cross-validation is used for measuring the predictive ability of a model as part of the model's evaluation. In this method, the data is partitioned into two sets such that one partition is used for method training and the other partition is used for testing. The part of the data that is to be used for actual testing is treated as new data and is used to test the performance of the resulting model. In our case this is done by computing the MCC.

   The reason we chose to perform cross-validation over residual evaluation is to avoid the issue of *overfitting*. *Overfitting* occurs when the model performs badly at making predictions, by yielding overly optimistic results that are not really correct. This happens because the model fits the data too closely, and as a result does not perform well on new data.

4. *Area Under the Curve (AUC)*

   The AUC of the ROC curve, which corresponds to the value of the Wilcoxon-Mann-Whitney test, is used as *"a measure of goodness for predictions"* [Vihinen, 2012]. An area of value 0.5 indicates a random classification while a value of 1 implies a perfect classifier. Therefore, in our algorithm evaluation, the preferred area value should be greater than 0.5 and ideally close to 1.

5. *Standard Error*

[4]ROCR: https://rocr.bioinf.mpi-sb.mpg.de/

Standard Error is a way of measuring the statistical accuracy of predictions by calculating by how much predictions vary from one another [std, 2016]. To calculate the standard error when running the algorithm in standalone mode and with GABI, we use the *anaesthetist*[5] webpage. To calculate the standard error for each network, with n nodes, in the DREAM4 datasets we use: the Area under the Curve, the number of edges in the gold standard network given by X and the non-edges given by $Y = (n^2 - n) - X$.

Next, to compare the two curves, derived from running the algorithm alone and in conjunction with GABI, we calculate the standard error of the difference between the two areas using the following equation.

$SE(A1 - A2) = \sqrt{SE^2(A1) + SE^2(A2)}$ [cal, 2016]

After the standard error of the difference in areas has been calculated for each network, we use that value to calculate the Z-score.

6. *Z-score*

   Z-score, also known as standard score, indicates how many standard deviations from the mean an element is, (either above or below the mean). Z-score provides us with a way of comparing two scores that belong to two different normal distributions. In our case from running the algorithm without GABI and from running it with GABI.

   The following equation derived from [cal, 2016] is used to calculate the Z-score.

$$Z = (A1 - A2)/SE(A1 - A2)$$

7. *P-value*

   We use the Z-score to calculate the two-tailed probability values (p-value), using the *Z-Score to P-Value Calculator* from [pva, 2016]. The P-value, or calculated probability, gives an indication of how strongly data supports or rejects a particular null hypothesis. The P-value can also be seen as the strength of evidence against the null hypothesis. Two-tailed P-value is the term used when the P-value is derived from both ends (tails) of the distribution [Ludbrook, 2013].

Now we proceed with the actual steps of the algorithm's evaluation and present the results.

5.1.2 DREAM4 runs

The HDP-HMM-DBN algorithm is evaluated in two modes: as a standalone and with GABI as a prior. Several runs are executed for each mode, (listed in the tables below), and the argument for the *maximum number of gene parents* (*NumParents*) is varied in

[5] Standard Error calculation: http://www.anaesthetist.com/mnm/stats/roc/Findex.htm

order to enable assessment of the effect of these parameters on the network structural influence as well as the algorithm's performance.

Runs - Insilico size 10	
Mode	NumParents
Standalone	3, 4, 5
With GABI	3, 4, 5

Runs - Insilico size 100	
Mode	NumParents
Standalone	3, 5, 7, 10
With GABI	3, 5, 7, 10

Table 5.1: Runs performed as part of the benchmarking process

The output of the HDP-HMM-DBN algorithm is an adjacency matrix containing the probability of each edge existing in the network. In the next step, the adjacency matrix for each network from the algorithm is compared to the adjacency matrix of the gold standard network.

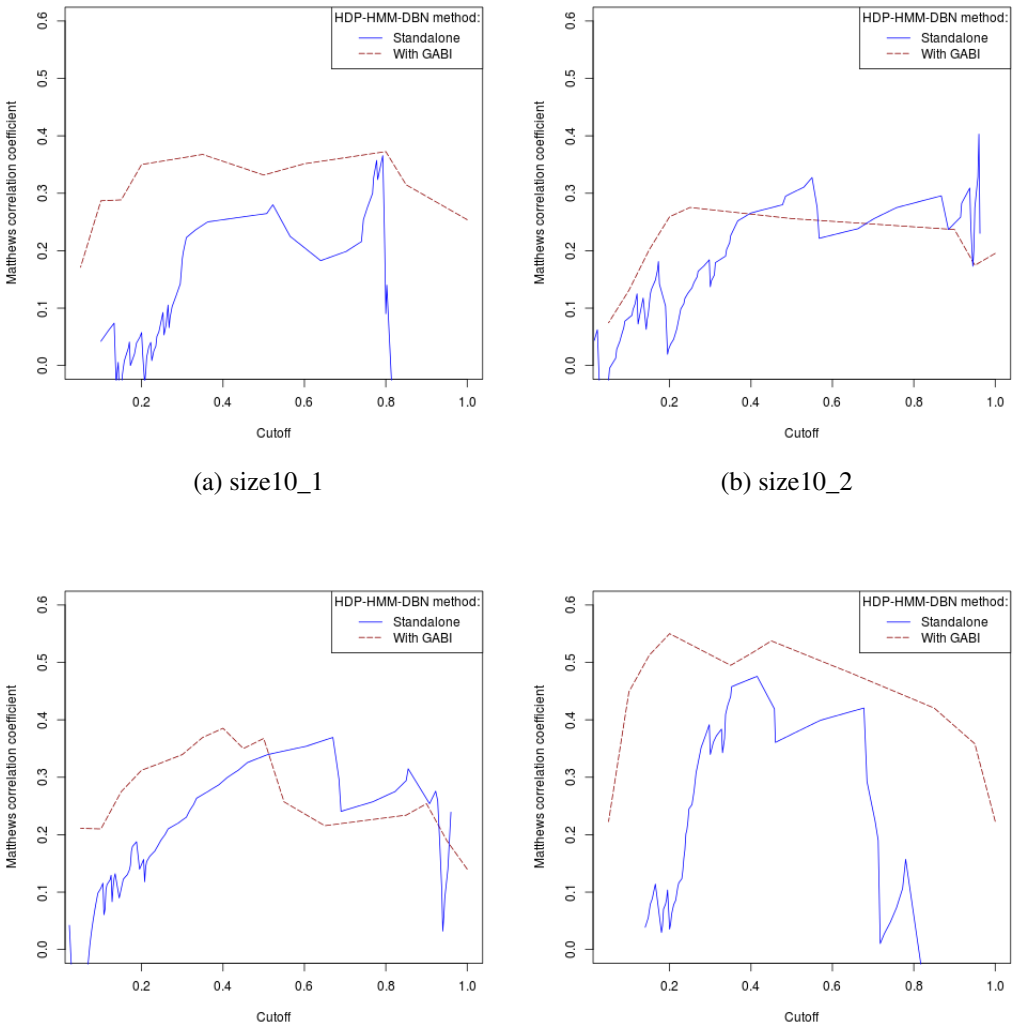
5.1.3 Performance against gold standard networks

The ROC curves and MCC graphs are generated using ROCR. This gives a measure of how the networks inferred by our algorithm, both in standalone and with GABI, compare to the gold standard networks.

For each network in the sub-challenge we use the adjacency matrix produced by our algorithm and test the predictions contained in it using ROCR. Those predictions are the estimated probabilities for the true values (labels) in the gold standard network. The list of predictions made by our algorithm is next passed to the *prediction* function of ROCR, along with the labels of the gold standard network. This gives us a *prediction* object which we next use as an argument in the *performance* method, along with the measure we are interested in. Four types of performance measure have been generated, namely: accuracy, precision-recall, MCC and true positive rate v/s false positive rate. The measure which we focus on is the MCC.

The MCC is calculated over the range of possible cutoffs by ROCR. The *cutoff* determines which of the edges found in the adjacency matrix, produced as output by the HDP-HMM-DBN algorithm, are considered as positive predictions. For example, if the cutoff is 0.8, we consider edges with probability >= 0.8 as positive edges.

The MCC plots for insilico size 10 ($numParents = 5$) and insilico size 100 ($numParents = 10$) are illustrated in Figures 5.1 and 5.2.



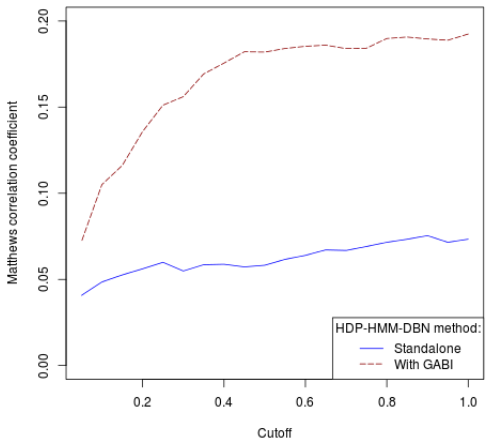
(a) size10_1

(b) size10_2

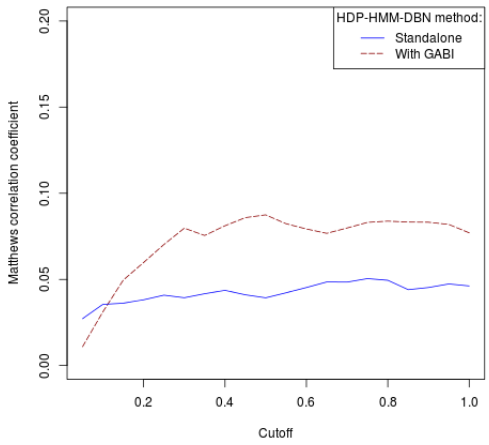
(c) size10_3

(d) size10_4

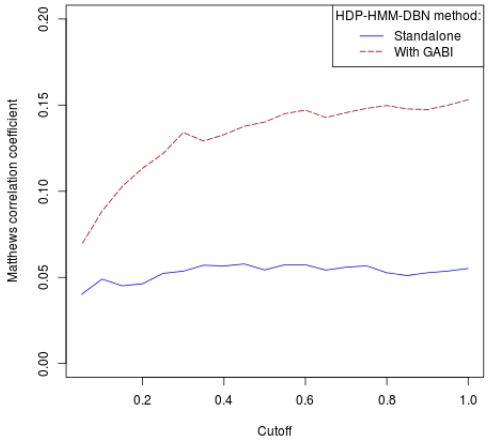
Figure 5.1: Algorithm's performance shown at different cutoff values using **insilico size 10** *datasets and* **numParents=5**. *The y-axis corresponds to the MCC and the x-axis refers to the cutoff value. Each graph shows the algorithm's performance when run on its own and in combination with GABI, depicted by a blue line and a red dashed line respectively.*



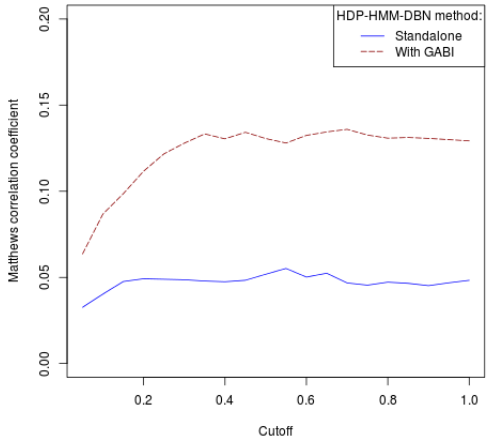
(a) size100_1



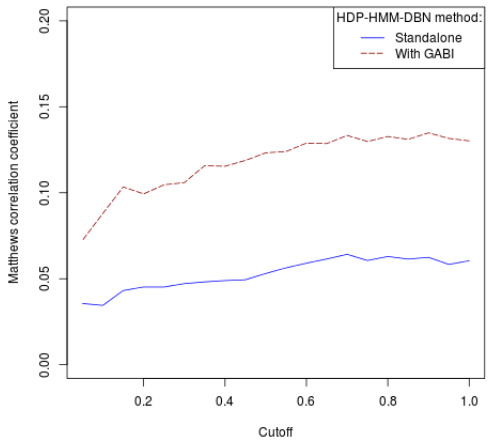
(b) size100_2



(c) size100_3



(d) size100_4



(e) size100_5

Figure 5.2: Algorithm's performance shown at different cutoff values using **insilico size 100** *datasets and* **numParents=10**. *The y-axis corresponds to the MCC and the x-axis refers to the cutoff value. Each graph shows the algorithm's performance when run on its own and in combination with GABI, depicted by a blue line and a red dashed line respectively.*

Figure 5.1 shows an irregular variation of the correlation coefficients for both standalone and with GABI which may be attributed to the small size 10 networks. This irregularity is particularly prominent for cutoff values greater than 0.5, indicating a low threshold for positive predictions using the small networks. Another reason might be that when the cutoff gets high, there are very few (or no) positive predictions made, and so the MCC decreases.

In the *size10_4* graph, the algorithm with GABI performs better, denoted by higher MCC values at each cutoff, where the cutoff value represents probability of the edge in the network. This can be verified by calculating the AUC for the HDP-HMM-DBN algorithm in both modes for network 4. Based on the values in Table 5.5, for network 4, the AUC with GABI is indeed greater than that without GABI. The difference between the AUC is even more prominent for network 1, with AUC without GABI being 0.6 and AUC with GABI being 0.7. Overall, GABI seems to improve the performance of the HDP-HMM-DBN algorithm for the insilico size 10 datasets, although the differences in results are not statistically significant (based on the P-values in Table 5.7)

Using the size 100 datasets it is clearly noticeable how using GABI as a prior with the HDP-HMM-DBN algorithm, gives better predictions, as indicated by a higher value for the MCC measure. These results are statistically significant as indicated by the P-values (0.0136, 0.0333, 0.028, 0.0119) for networks 1, 3, 4 and 5. Based on Table 5.6, we can see that the AUC for the algorithm with GABI is greater that that without GABI, with regards to all networks. Additionally, the AUC is greater when the maximum number of parents is equal to the number of replicates (10 in this case) in the data, compared to maximum number of parents being smaller (See Table 5.5 and 5.6). In particular, with increasing cutoff values, we have improved correlation coefficients in general for all networks.

5.1.4 Cross-Validation

The output of the HDP-HMM-DBN algorithm is an adjacency matrix with the probability of each edge existing and the cutoff determines which ones we take as positive predictions. The aim for performing cross-validation is to find the optimal cutoff value corresponding with the maximum value for MCC.

In our case, the insilico size 10 data is composed of 5 files. We select only 4 of them to be used as training set and keep the remaining one for testing. We calculate the MCC performance measures for each of the 4 training files. The coefficients at each cutoff are then averaged. From these values we find the maximum average MCC value and its corresponding cutoff value. This step is done to find the optimal cutoff value which we then apply to the test file to derive the corresponding MCC. As we have 5 data files the process is repeated 5 times so that each of the 5 files may be treated as test data; the other files being used as training data as required.

Cross-validation has been performed for all the results generated by the runs listed in Table 5.1. A few of the cross-validations tables generated are listed below with the MCC measures. Probabilities above cutoff value indicate positive predictions.

		Standalone		*With GABI*	
Train	Test	Optimal Cutoff	MCC	Optimal Cutoff	MCC
Insilico size 10, NumParents = 3					
2,3,4,5	1	0.32	0.1731992	0.30	0.3540992
1,3,4,5	2	0.32	0.1896601	0.30	0.2699408
1,2,4,5	3	0.58	0.1276440	0.30	0.2362457
1,2,3,5	4	0.44	0.2853667	0.35	0.4583358
1,2,3,4	5	0.40	0.3104287	0.30	0.3415879
Insilico size 10, NumParents = 5					
2,3,4,5	1	0.52	0.2774996	0.38	0.3604735
1,3,4,5	2	0.52	0.3072408	0.38	0.2652605
1,2,4,5	3	0.52	0.3410815	0.38	0.3789498
1,2,3,5	4	0.52	0.3817690	0.38	0.5072600
1,2,3,4	5	0.55	0.3916288	0.40	0.3852757

Table 5.2: Cross-validation table for **insilico size 10** *and* **numParents = 3 and 5**.

		Standalone		*With GABI*	
Train	Test	Optimal Cutoff	MCC	Optimal Cutoff	MCC
Insilico size 100, NumParents = 3					
2,3,4,5	1	0.20	0.01825332	0.45	0.03550667
1,3,4,5	2	0.20	0.03168943	0.45	0.01904052
1,2,4,5	3	0.25	0.03013492	0.45	0.04197943
1,2,3,5	4	0.60	0.02097569	0.45	0.04175986
1,2,3,4	5	0.20	0.02971240	0.45	0.03590798
Insilico size 100, NumParents = 10					
2,3,4,5	1	0.65	0.06712193	0.80	0.18984800
1,3,4,5	2	1.00	0.04623111	1.00	0.07704713
1,2,4,5	3	0.80	0.05273394	0.90	0.14744300
1,2,3,5	4	0.75	0.04550241	0.80	0.13083250
1,2,3,4	5	1.00	0.06048800	0.80	0.13273450

Table 5.3: Cross-validation table for **insilico size 100** *and* **numParents = 3 and 10**.

	With GABI	
Test	MCC	MCC
	Insilico size 10	*Insilico size 100*
1	0.23924690	0.006519519
2	0.32732680	0.089337880
3	-0.03474072	0.111623900
4	0.20567190	0.068116080
5	Empty network returned	0.071879000

Table 5.4: Baseline performance table for GABI only

5.1.5 Area under the ROC curve

We calculate the area under the curve when comparing the results of the runs with the gold standard network. These values are used later on to derive: the standard error, Z-Score and two-tailed P-value for each network in the size 10 and size 100 sub-challenges.

The tables listing the area under the curve for insilico size 10 *(numParents=3, 5)* and insilico size 100 *(numParents=5, 7, 10)* are given below.

	NumParents = 3		*NumParents = 5*	
size10 Network	*Standalone* AUC	*With GABI* AUC	*Standalone* AUC	*With GABI* AUC
1	0.6117647	0.6807843	0.6003922	0.7015686
2	0.6700149	0.6741071	0.6536458	0.6082589
3	0.6207843	0.6592157	0.6796078	0.7113725
4	0.7427056	**0.7763042**	**0.7701149**	**0.7979664**
5	**0.7902462**	0.7902462[a]	**0.7940341**	0.7940341[b]

Table 5.5: Table listing the AUC for **insilico size 10** *and* **NumParents = 3, 5***. AUC values greater than 0.75 are in bold face indicating a good binary classifier.*

[a]Note: No network returned by GABI
[b]Note: No network returned by GABI

	NumParents = 5		*NumParents = 7*		*NumParents = 10*	
size100 Network	*Standalone* AUC	*With GABI* AUC	*Standalone* AUC	*With GABI* AUC	*Standalone* AUC	*With GABI* AUC
1	0.5093391	0.5447550	0.5402797	0.5881821	0.5746433	**0.6538027**
2	0.5108880	0.5276248	0.5389281	0.5565336	0.5412943	0.5316289
3	0.5328762	0.5435602	0.5226067	0.5654760	0.5669465	**0.6319084**
4	0.5354981	0.5662960	0.5260552	0.5571204	0.5536849	**0.6181007**
5	0.5141168	0.5639231	0.5739836	0.5706252	0.5592627	**0.6362945**

Table 5.6: Table listing the AUC for **insilico size 100** *and* **NumParents = 5, 7, 10***. AUC values greater than 0.6 highlighting the best performance are in bold face.*

The standard error, Z-score and P-value for each network in the insilico size 10 and size 100 datasets, are given in Table 5.7. For insilico size 10, standard errors are given only for the first three networks because the X value derived for networks 4 and 5 are not within the accepted range of values in the "Calculate Standard Error" form on the anaesthetist webpage.

Network	DREAM sub-challenge	NumParents	AUC (with GABI)	AUC (Standalone)	Z-Score	Two-tailed P-value
1	10	5	0.7015686	0.6003922	0.8724642724	0.383
2	10	5	0.6082589	0.6536458	-0.3972860858	0.6912
3	10	5	0.7113725	0.6796078	0.278449767	0.7807
4	10	5	0.7979664	0.7701149	0.0758[a]	0.9396
5	10	5	0.7940341	0.7940341	No network returned by GABI	
1	100	5	0.544755	0.5093391	1.1242138502	0.2609
2	100	5	0.5276248	0.510888	0.6318524733	0.5275
3	100	5	0.5435602	0.5328762	0.3546811284	0.7228
4	100	5	0.566296	0.5354981	1.0579093567	0.2901
5	100	5	0.5639231	0.5141168	1.6467815434	*0.0996
1	100	10	0.6538027	0.5746433	2.4685401179	*0.0136
2	100	10	0.5316289	0.5412943	-0.3618987633	0.7174
3	100	10	0.6319084	0.5669465	2.1285830613	*0.0333
4	100	10	0.6181007	0.5536849	2.1977449668	*0.028
5	100	10	0.6362945	0.5592627	2.5147431498	*0.0119

Table 5.7: Standard error, Z-score and P-value

[a]Underestimate value: Slight underestimate due to form not accepting values < 15. However the bottom-lying results of this comparison shows no significant difference.

5.1.6 Integrated analysis

In this step, we further illustrate the MCC performance measures of the HDP-HMM-DBN algorithm based on results of the different runs of the method with and without GABI and with varying number of gene parents. Further graphical representations of runs pertaining to insilico size 10 and size 100 datasets are shown in Figure 5.3 and Figure 5.4. These graphs show how the performance of the HDP-HMM-DBN algorithm is influenced by: varying the maximum number of gene parents, and using GABI.

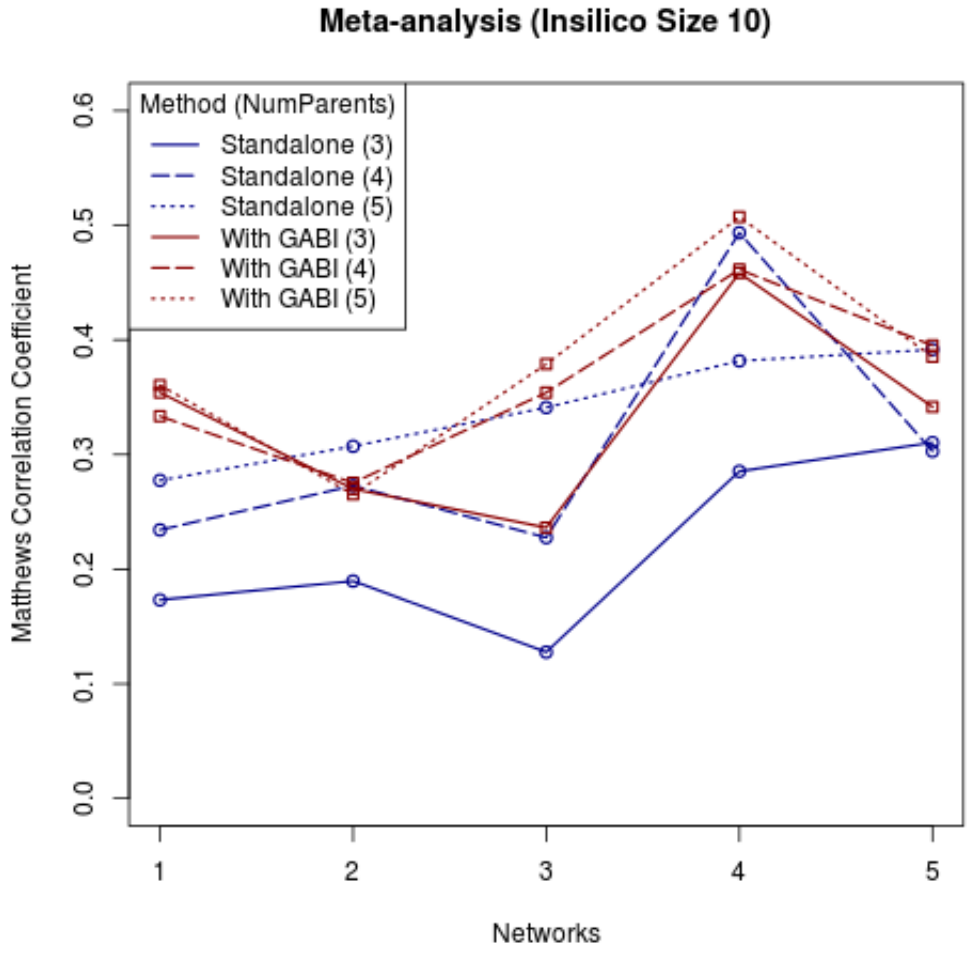


Figure 5.3: MCC performance measures of the HDP-HMM-DBN method shown for each network in the insilico size 10 datasets with varying number of gene parents. The x-axis refers to the networks in the size 10 datasets and the y-axis refers to the MCC. The blue lines represent the method's performance without GABI and the red lines correspond to the method's performance when used in combination with GABI. The different types of line correspond to the different number of gene parents.

Based on the meta-analysis graph, (Figure 5.3), illustrating the runs using insilico size 10 data, we can deduce that the algorithm overall performs better, both with and without GABI, when the maximum potential number of gene parents is set to the number of replicates in the gene expression data (5 in this case). The largest possible value we can allocate to the variable, for the maximum number of gene parents, is limited to the number of replicates. The algorithm performs better when the maximum number of gene parents is larger because in the real data, there may be nodes with many parents.

To verify how the algorithm's performance is affected by the value of the maximum potential number of gene parents, we use the AUC values from Table 5.5 for $NumParents = 3$ and 5. For networks 3, 4 and 5, the AUC for the HDP-HMM-DBN algorithm, in both modes, has a higher value when the number of gene parents equals 5 as opposed to with number of gene parents equals 3. Network 1 also has a higher AUC when the method is run in conjunction with GABI and $NumParents = 5$. Another interesting point, for the standalone version, is that the cutoff with $NumParents = 5$ seems more stable than with $NumParents = 3$ (Table 5.5). Additionally, performance is improved considerably in the standalone and slightly with GABI as prior as indicated by the higher MCC values.

We also note that the MCC is higher for most networks when the algorithm is run in conjunction with GABI. This can checked using the AUC tables for $NumParents = 3$ and $NumParents = 5$ (Table 5.5). In both tables, the AUC when the method is used with GABI is greater than the AUC when the method is used without GABI for most of the networks.

However, GABI does not always improve performance, for example, predictions for network 2 , where $NumParents = 5$, are more accurate with the HDP-HMM-DBN algorithm run in standalone mode. This can be verified using Table 5.5, where the AUC for the method run without GABI is 0.6536 while the AUC for the method run in combination with GABI is 0.6083.

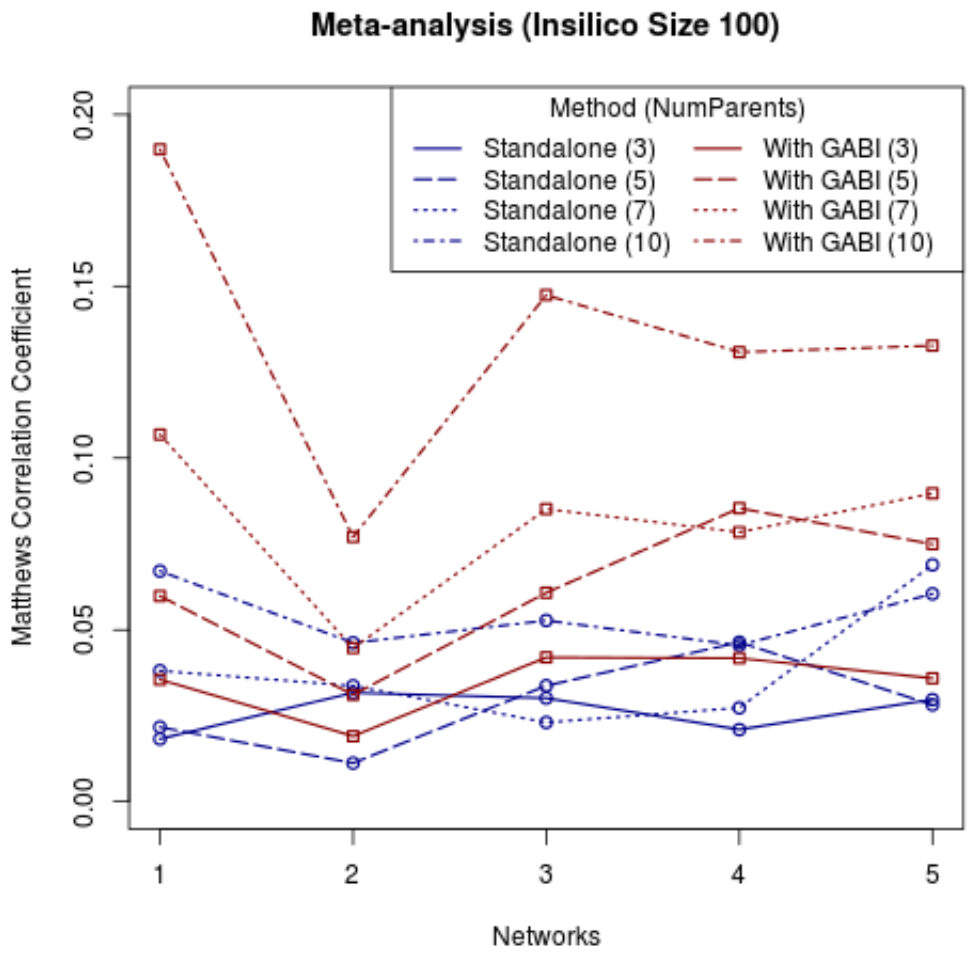


Figure 5.4: MCC performance measures of the HDP-HMM-DBN method shown for each network in the insilico size 100 datasets with varying number of gene parents. The x-axis refers to the networks in the size 100 datasets and the y-axis refers to the MCC. The blue lines represent the method's performance without GABI and the red lines correspond to the method's performance when used in combination with GABI. The different types of line correspond to the different number of gene parents.

Similarly, for the insilico size 100 runs, better performance is achieved when the highest potential number of parents equals to the number of replicates in the time series data used as input. This can be verified using the AUC when the $NumParents = 5$ (Table 5.5) and the $NumParents = 10$ (Table 5.6). For all 5 networks, both with and without GABI, the AUC is significantly higher when the number of gene parents is equal to 10, as opposed to the number of gene parents equal to 5.

Additionally, when GABI is used as prior, we note that the MCC performance measure is higher, denoted by the red dashed-dotted line in the graph. Overall, the AUC for the algorithm run with GABI is bigger than that run without GABI as shown in Tables 5.5 and 5.6. GABI has a statistically significant improvement in the performance of the HDP-HMM-DBN algorithm in many cases, as indicated by the P-values in Ta-

ble 5.7. With reference to the insilico size 100 sub-challenge with $NumParents = 10$, for networks 1, 3, 4 and 5, we note that the algorithm when used in conjunction with GABI gives better results, denoted by the following P-values: 0.0136, 0.0333, 0.028 and 0.0119.

Based on Table 5.4, GABI on its own gives better overall performance using the insilico size 100 datasets, denoted by higher MCC values, than the HDP-HMM-DBN algorithm in standalone mode (Table 5.3). However, for the insilico size 10 datasets, the HDP-HMM-DBN algorithm, in standalone mode, gives better predictions when the maximum number of gene parents is equal to 5 (Table 5.2.

The meta-analysis graphs and the AUC and statistical analysis tables for both insilico size 10 and size 100 results lead us to conclude that the HDP-HMM-DBN algorithm gives better predictions when used in combination with GABI and when the maximum possible number of gene parents is equal to the number of replicates in the expression data. Running the algorithm on the renal cancer data, and armed with this information, we set the *numParents* argument to be equal to the number of replicates and used GABI as a prior.

Before running the HDP-HMM-DBN algorithm on the renal cancer data, we verify the change in network structure at each timepoint using time series data for network 1 and 5 from the insilico size 10 datasets.

5.1.7 Validating the network structure change using DREAM4 data

To verify the change in network topology at each timepoint in the gene expression data, we concatenate the datasets for two networks, namely networks 1 and 5. We then run the HDP-HMM-DBN algorithm on it. As mentioned previously, for the first half of the time series data, a perturbation was added at time *t=0* and then it was removed halfway through the gene expression data at time *t=500*. In the second half of the data (*t=500* to *t=1000*), the network relaxes.

The time steps for the concatenated (interleaved) datasets of networks 1 and 5 are to be interpreted as follows:

- the first quarter corresponds to network 1 with perturbation
- the second quarter corresponds to network 1 with perturbation removed
- the third quarter corresponds to network 5 with perturbation
- the fourth quarter corresponds to network 5 with perturbation removed

The output of the HDP-HMM-DBN algorithm in this test is a list of state sequences at each time point in the concatenated gene expression data (network 1 and 5). This list tells us which network we have at each time point. Using the state sequence, we derive a matrix containing 1 or 0, depending on whether the network is the same between each possible pair of timepoints. A visualisation of the matrix is shown in Figure 5.5 where the time on the x and y axis refers to time steps 1 to 41 corresponding to the actual time 0 to 1000 with interval 50.

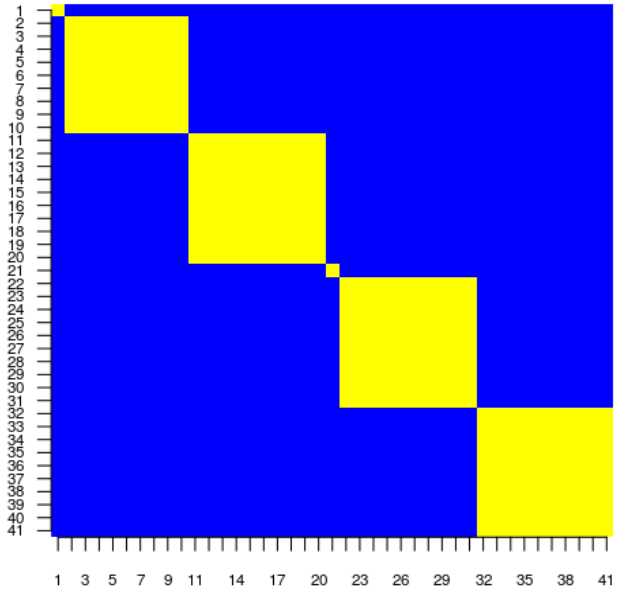


Figure 5.5: Network structure change using time series data for networks 1 and 5 from size 10 datasets

Values in the X and Y directions represent time points in the concatenated time series data for networks 1 and 5. Based on Figure 5.5, we can see that the network changes structure at timepoint 2, stays the same for the next 8 consecutive timepoints and then changes again at time point 11 and so on.

Based on these results, we can see that although it is not perfect (there are networks at time 1 and 21 that are only for that time point) it appears to correspond to the changes we might expect for that data (the way it is divided up into quarters, listed above), and so the time changing part is working.

5.2 Renal Cancer data

The HDP-HMM-DBN method is applied on renal cancer time course transcriptome data from four representative cell lines exposed to the drug Sunitinib. A cell line is *a clone of cultured cells derived from an identified parental cell type* [cel, 2016b].

Four cell lines were obtained from human cancer tissues (metastasis, primary tumours) or endothelium, and were chosen from a panel of sixteen cell lines (Overton, personal communication). These lines will enable us in the investigation of the mechanisms involved in the spread of cancer cells to distant sites, and drug response/resistance and angiogenesis in the context of Sunitinib treatment. The data consists of six time points following Suntitib exposure in two conditions – hypoxia and normoxia.

Hypoxia is a term used in a medical context to indicate a condition where the tissues are deprived of oxygen. It is often found in the central region of tumours (referred as tumour hypoxia) due to the lack of vascularisation in those particular areas. Tumour

hypoxia is increasingly recognised as a detrimental factor in cancer therapies. The lack of adequate oxygen in the tissues can have a negative impact on treatment and can aid malignant tumour growth [McKeown, 2014]. Normoxia is the term used to indicate "normal" oxygen levels in tissues.

Work in this thesis will examine the time course drug response data from one of these lines (Caki-1).

5.2.1 Drug response and resistance

The renal cancer data has been provided by Hans-Joachim Sonntag, a PhD student from the Overton lab. This data is derived from the drug resistant Caki-1 cell line which has been cultured under hypoxic conditions. Caki-1 cells were originally obtained from a human renal cell carcinoma [Schömig and Schönfeld, 1990], and as a result they represent a useful model in the study of renal cancer.

The data is provided in the form of three text files, each file containing data for one replicate. Each row in the file consists of a gene identifier (an entrezID[6] in our case), followed by tab-separated gene expression values for six time points. Before running the algorithm on the cancer data files the data is converted into the required format: genes presented column-wise and rows representing the different time steps. Rows with the same time stamp, representing replicates, are arranged to be consecutive. Gene expressions with zero values are replaced by some Gaussian noise to allow for the correct functioning of the HDP-HMM-DBN algorithm. The reason is that because if multiple observations for a specific gene are exactly the same number, whether it is 0 or some other value, there is not enough information to calculate the likelihood. Therefore, adding some small amount of noise means that they are not exactly the same, and so the method will run.

The algorithm is launched once the data is in the correct format, GABI is enabled and the variable *numParents* is set to three (the number of replicates in the renal cancer data).

The adjacency matrix, the output of our algorithm, is next converted into a weighted adjacency list in order to to be compatible with Cytoscape.

Cytoscape[7] is a software platform used for the integration of molecular interaction networks with high-throughput gene expression profiles and data. It provides an intuitive way of visualising and querying the network by, for example, linking it to databases of functional annotations. The functionality of Cytoscape is further enhanced by plug-ins, which can be used to perform additional analysis of the molecular networks. The *BiNGO* plugin can be added to Cytoscape to find which Gene Ontology[8] (GO) terms are more represented in a set of genes found in a particular biological network.

[6]Entrez Gene - NCBI's database for gene-specific information: http://www.ncbi.nlm.nih.gov/gene
[7]Cytoscape: http://www.cytoscape.org/what_is_cytoscape.html
[8]Gene Ontology: https://www.ebi.ac.uk/QuickGO/

Cross-validation on the DREAM4 gold standard 100 node network sub-challenge data identified a threshold of 0.45 (Table 5.3). The insilico size 100 sub-challenge is chosen to determine the threshold value for the renal cancer data since the size of the networks contained therein matches the size of the CAKI1 input dataset. This threshold value is used to prune insignificant edges and ascertain that only highly significant edges, with a weight of 0.45 or higher, are kept. By removing edges with a low weight, only a subset of genes from the renal cancer data are used to generate the network in Cytoscape.

Once the weighted adjacency list for the *ApoCluster_CAKI1_Hyp* data is ready, it is imported into Cytoscape and a directed graph depicting the gene interactions in the renal cancer data is generated. This network is shown in Figure 5.6 .

It is helpful, for further biological investigation of the gene regulatory network inferred from the renal cancer data, to be able to tell from the network edges, which of the genes are inhibitory and which ones increase the transcription of the genes they are pointing to. We use Spearman's Rank Correlation Coefficient for this.

Spearman's Rank Correlation Coefficient is a statistical method used to detect monotonic relationships. It helps verify the strength, direction and sign (positive or negative) of a relationship between two variables. This technique is normally applied when evaluating the truth of a hypothesis. The coefficient value is between -1 and +1. The closer the coefficient value is to these extremes, the stronger the correlation is deemed to be. In our application, a Spearman Corrleation matrix is created from the weighted adjacency list of genes used to create the initial network. This matrix contains the values of the Spearman Correlation Coefficient for each edge in the network and indicates the strength and sign of the link between pairs of genes. After the Spearman Correlation matrix is created, it is loaded into *Cytoscape* to give a sign to each edge in the network. The resulting network is displayed in Figure 5.6.

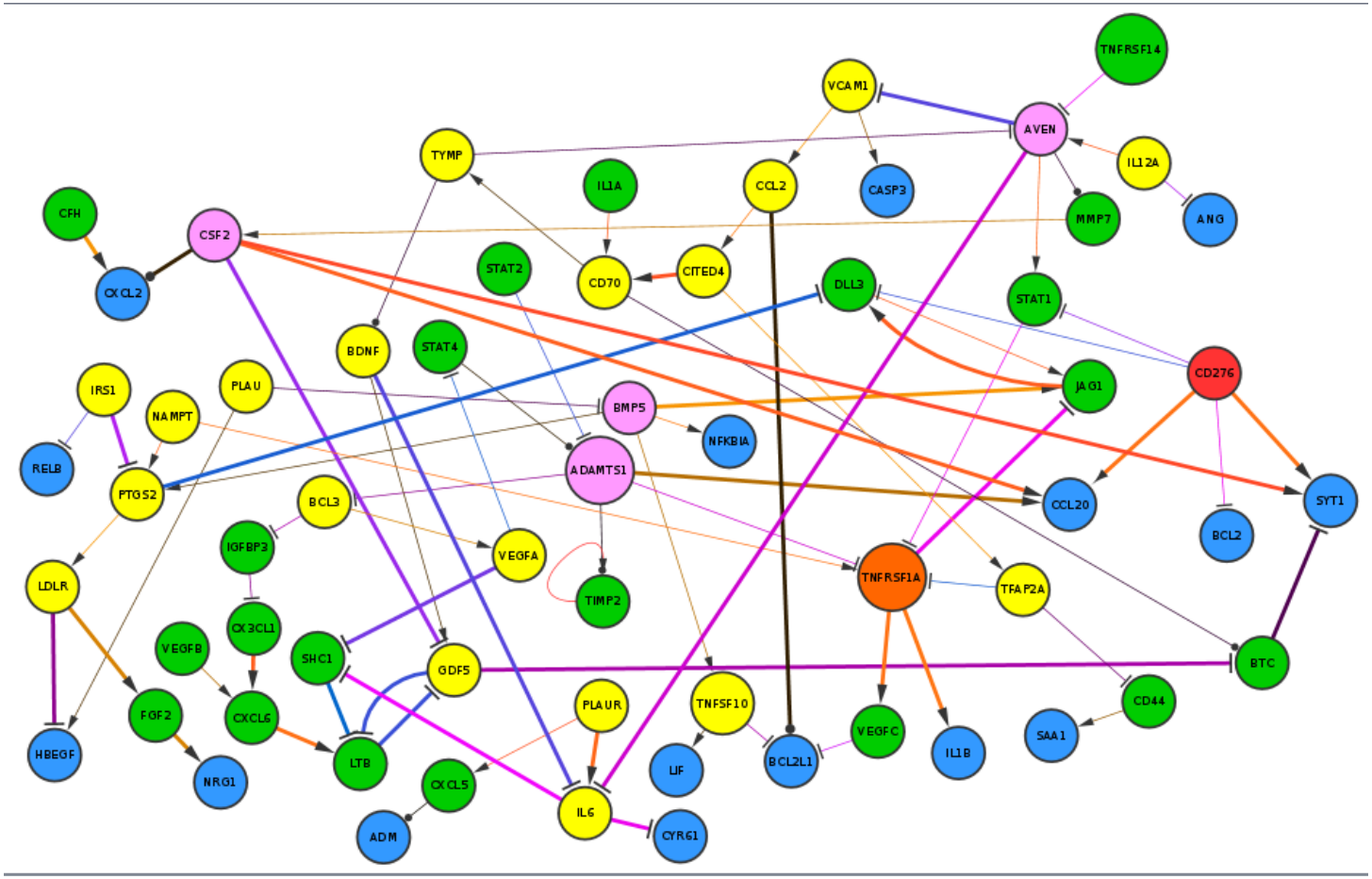


Figure 5.6: Network generated for the 'ApoCluster_CAKI1_Hyp' renal cancer data. Each node represents a gene. The colour of the node indicates the out-degree value and the edges represent the regulatory relationships between the genes. The sign of the edge represents the type of relationship (activatory or inhibitory) between the pair of genes

Explanation of Figure 5.6

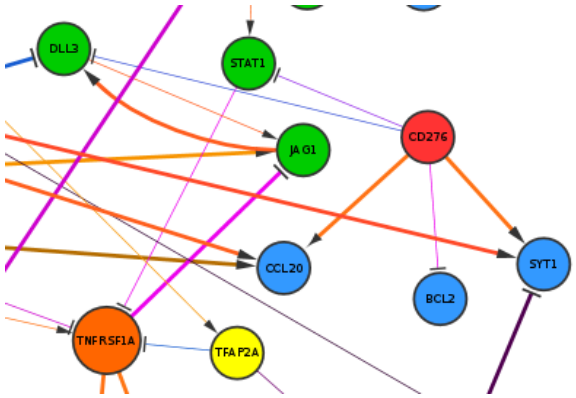


Figure 5.7: Each node represents a gene. The colour of the node indicates the out-degree value and the edges represent the regulatory relationships between the genes. The sign of the edge represents the type of relationship (activatory or inhibitory) between the pair of genes

The Uniprot[9] function for each of the genes mentioned in the Analysis chapter is given in Appendix A.

The colour of a node is based on its outdegree. Outdegree is the number of edges emanating from a node and is the number of child nodes. Node CD276, for example, has five edges emanating from it and is thus coloured red; red nodes have five children. Similarly: pink, orange, yellow, green and blue represent genes having: four, three, two, one and zero children respectively.

Edges have three attributes: colour, sign and thickness.

The colour of an edge is based on the Spearman Correlation value derived from the Spearman Corrleation matrix. A colour gradient with colours starting from blue, pink, black, orange and red are used to represent the correlation values between -1 and 1. Blue lines indicate a Spearman correlation in between -1 and -0.4, pink lines represent a correlation between -0.4 and 0. Black lines represent a correlation approximating 0. Orange lines refer to correlation value between 0 and 0.45 and red represent lines with correlation approximating 1.

The edge sign indicates the relationship between a pair of genes, as defined by the Spearman Correlation strength. $\rightarrow$ implies activation and $\dashv$ implies inhibition. There is also a non-linear relationship where, for example, the gene may exhibit inhibition at low or high concentrations, but activation at a middle concentration. A non-linear relationship is represented by a dot at one end of an edge, as shown with the gene CSF2, (pink node at the top left of the network). The gene CD276 (shown in the network detail, Figure 5.7) has the highest outdegree and seems to activate CCL20 and SYT1 but inhibits the expression of BCL2, DLL3 and STAT1.

The thickness of an edge represents the probability of the edge, as predicted by the HDP-HMM-DBN algorithm, appearing in the gold-standard network. Thicker edges are predicted as having a higher probability of existing.

[9]Uniprot Website: http://www.uniprot.org/

The BCL2 gene is an apoptosis regulator in humans. As mentioned previously, apoptosis is a process whereby cell death is programmed for cells which have reached the end of their useful life. BCL2, a protein-coding gene, suppresses apoptosis in certain cells such as lymphocytes, (white blood cells forming part of the immune system), and neural cells [bcl, 2016]. CD276 is a protein-coding gene found in humans. This gene is involved in controlling the T-cell-mediated immune response and may also be related to tumour cells [CD2, 2016]. It may be of interest that our regulatory network for the renal cancer data indicates a possible inhibitory role for the C276 gene with regard to BCL2.

BCL2L1 is another gene which plays a key role in apoptosis. It encodes a protein related to the BCL2 protein family. The proteins encoded by this gene are involved in the inhibition of cell death [BCL, 2016]. Our gene regulatory network indicates a non-linear role for CCL2 with regard to BCL2L1 and it is shown with maximum weight.

The activation of JAG1 by BMP5, which leads to the establishment of notch signalling, is already an established biological fact [Zavadil et al., 2004; Niimi et al., 2007]. It is encouraging that this relationship is shown on our gene regulatory network with a strong probability.

The relationship between VEGFC and BCL2L1 is also interesting because hypoxic signalling (VEGFC) is predicted to confer resistance to apoptosis (BCL2L1), this overall relationship is also consistent with literature where resistance to hypoxic stress is expected to include evasion of apoptosis [Greijer and Van der Wall, 2004]

The following highlights some prominent topological features exhibited by the network:

- CD276 has the most influence on other gene expressions with the highest out-degree.
- With the highest in-degree, TNFRSF1A is the most influenced, including from CD276 through STAT1.
- TIMP2 is self regulating its expression.
- A positive feedback exits between: LTB and GDF5; DLL3 and JAG1.
- BCL3 influences LTB through IGFBP3-CX3CL1-CXCL6 and VEGFA-SHC1.

To understand the biological functions of the genes in the network, DAVID[10] (Database for Annotation, Visualization and Integrated Discovery), is used to identify gene pathways and gene ontologies. DAVID was chosen because of its ease of use and range of functionality. For example, with DAVID, we can identify the biological themes related to the genes of interest and also find functionally related gene groups. As part of our analysis we imported all the genes in the network into DAVID and investigated which pathways, disease and functional categories the genes are associated with. The associations are highly significant, for example the top result has a Benjamini corrected

[10]DAVID: https://david.ncifcrf.gov/

P-value of 6.1E-12 and is associated with Cytokine-cytokine receptor interaction. Cytokines are inflammatory mediators present in tumor tissues [Mantovani et al., 2008]. Some of the KEGG pathways involve apoptosis and chemokine signaling, which are linked to cancer.

The top results for the gene ontologies are related to regulation of cell proliferation, regulation of programmed cell death, regulation of apoptosis, chemotaxis and negative regulation of apoptosis.

Chapter 6

Conclusion

6.1 Summary

This thesis has proposed a new method to reverse engineer gene regulatory networks from gene expression data. The method uses the Infinite Hidden Markov model and a Dynamic Bayesian Network (DBN) model to represent the predicted gene networks. We also investigate the use of GABI, a relevance thresholding algorithm developed by the Overton group, as a prior to our method. Our method allows us to predict gene networks that can change their structures at different time points. To model the sequential nature of time series data, it uses a non-parametric framework which can be modified as necessary to adjust to the complexity of the data. This non-parametric framework is the Hierarchical Dirichlet Process-Hidden Markov Model (HDP-HMM), an extension of the standard Hidden Markov Model (HMM). Unlike the HMM, where the number of hidden states is known in advance, the HDP-HMM can have an infinite number of hidden states depending on the data.

In the HDP-HMM model, a Dirichlet Process prior is used to calculate the transition probabilities of moving from one hidden state to another. The distributions associated with the individual states are grouped in a hierarchical structure, which facilitates transitions between the potential states. In the original method [Thorne and Stumpf, 2012], a Bayesian network is used for the network representation. In this thesis, to overcome the limitation of acyclic graph associated with Bayesian networks, we extend the original method by using Dynamic Bayesian networks to model the gene regulatory networks at each hidden state. Our extended method is referred to as the HDP-HMM-DBN method.

When deriving DBN structures, the MH Sampler algorithm is used to sample the nodes for the network. This sampler can simulate distributions with a large number of dimensions such as gene expression data. The HDP-HMM-DBN method can be passed as argument a prior which removes edges with low significance in the network. We use GABI as prior in our method. GABI also predicts directionality using information-theory and the properties of the undirected relevance network.

We benchmark our method on data from the DREAM4 challenge, using time series

datasets from the Insilico_Size10 and Insilico_Size100 sub-challenges. Several runs were executed on the DREAM4 data with the number of gene parents as a variable. For each value of the gene parent variable, the run is repeated with and without GABI. The ROCR package is used to assess the performance of our method by comparing the networks inferred from these runs with gold standard networks. We generate ROC curves and graphs pertaining to the Matthews Correlation coefficient to visually analyse the method's performance. We also calculate areas under the ROC curves as part of our analysis which we use to derive the standard error and the two-tailed P values used for testing the differences in results when running the method with and without GABI.

Based on the Matthews Correlation Coefficient graphs generated, the method (with and without GABI) is shown to give an irregular variation of the correlation coefficient when run on the Insilico_Size10 datasets. The threshold for positive predictions is also quite low with a value of less than or equal to 0.5. This variability may be related to the small size of the networks. It is also known that GABI performs better with larger networks. Using the Insilico_Size100 datasets, the algorithm's performance is more uniform across the different networks. It is also more obvious from the graphs how GABI helps to give better predictions, indicated by a higher coefficient value and a larger cutoff number. From the meta-analysis graphs in Figure 5.3 and Figure 5.4, it is clearly visible that the HDP-HMM-DBN method gives more accurate predictions when the maximum number of potential parents of a node is larger than the number of replicates in the gene expression data, although in practice this number is limited by the data.

Based on the information derived from the benchmarking process, we make assumptions regarding the best settings for the HDP-HMM-DBN method which gives us optimal performance. We note that our method performs better when used in combination with GABI and when the parameter corresponding to the maximum number of gene parents is equal to the number of replicates in the data.

The HDP-HMM-DBN method is then run on the renal cancer data provided by the Overton group. We ensure that the method is run with GABI enabled and the parameter for the number of gene parents set to the number of replicates in the cancer data. Using the results produced by our method, we generate a regulatory network using Cytoscape, a platform for analysing and visualising regulatory networks. The Cytoscape network can be used for further biological analysis. For instance, gene interactions at specific points during apoptosis and angiogenesis can be studied.

6.2 Contributions

In this section, we give a summary of the main contributions of this thesis.

1. *Extension of the method proposed in [Thorne and Stumpf, 2012].*
   We implement a new method that builds on the one advanced in [Thorne and Stumpf, 2012] and improve it by representing gene regulatory networks using a dynamic Bayesian network model. The method has been developed in R and

made into a package which makes it portable and thus can easily be shared in the systems biology community.

2. *Integration with the approaches developed by the Overton group.*
   GABI, an algorithm designed for the inference of small-scale network based on tissue microarray (TMA) data, and developed by Alex Lubbock from the Overton group, is integrated with the HDP-HMM-DBN method. By using GABI as prior in our method, we ensure that only "high confidence" relationships between genes in the networks inferred by our method are kept. GABI removes edges which are below a certain relevance threshold value, determined by Spearman correlation and symmetric uncertainty. Additionally, the edges which result from GABI are signed or unsigned according to the Spearman correlation, and have directionality based on information-theory and the properties of the network. We show how overall performance is improved when the HDP-HMM-DBN method is used in combination with GABI.

3. *Dynamic analysis of biological data.*
   The HDP-HMM-DBN method can be used to generate time-varying gene regulatory networks from gene expression profiles. By using our method on biological data, such as renal cancer data, we can gain a better understanding of how genes interactions change with time and during certain events.

   The biomedical application of this project is to develop useful tools to assist in the understanding of the biology of diseases such as renal cell carcinoma, and how they respond to certain drugs. For example, use of our method on the renal cancer data indicates a highly probable non-linear role for CCL2 with regard to BCL2L1. BCL2L1 is a gene that plays a key role in apoptosis as it encodes proteins which are involved in the inhibition of cell death.

6.3 Limitations and Future Work

Network reverse engineering is a hard problem. This thesis has looked at integrating different methods and developing a benchmark that includes different network structures at different times (matching the aims of [Thorne and Stumpf, 2012]). There is much room for improving the method in terms of predictions and speed.

We plan to document the code and improve the user interface. This should make it easier for other developers and biologists to run the method on their own data. We would also like to support a wider range of input for the network. We plan to upload the source code of our implementation into GitHub as an R package in order to encourage its use and further development by other researchers.

To improve the algorithm's performance in terms of speed, certain parts of the code can be optimised by performing parallel programming through careful use of R libraries and packages that offer parallel versions of compute-intensive functions, for example graph operations such as search. Alternatively, the method could be developed to run in *Hadoop*, a framework for performing distributed processing of big data using

computer clusters. This improvement would make it possible to run the method on larger datasets, enabling analysis of results which is currently not feasible. Running experiments would also be less time-consuming.

This method is not necessarily restricted to analysing gene expression data. Another direction for further work is to target the generality and wider applicability of the method in other domains requiring similar modelling. For example, inferring biological neural networks from time series data, or modelling particle-based interactions in a physical system.

Appendix A

Uniprot genes function

The following is the Uniprot function for the genes mentioned in the Analysis chapter (source: `http://www.uniprot.org/`).

Gene Name: CD276
Protein: CD276 antigen
Function: May participate in the regulation of T-cell-mediated immune response. May play a protective role in tumor cells by inhibiting natural-killer mediated cell lysis as well as a role of marker for detection of neuroblastoma cells. May be involved in the development of acute and chronic transplant rejection and in the regulation of lymphocytic activity at mucosal surfaces. Could also play a key role in providing the placenta and fetus with a suitable immunological environment throughout pregnancy. Both isoform 1 and isoform 2 appear to be redundant in their ability to modulate CD4 T-cell responses. Isoform 2 is shown to enhance the induction of cytotoxic T-cells and selectively stimulates interferon gamma production in the presence of T-cell receptor signaling.

Gene Name: BCL2
Protein: Apoptosis regulator Bcl-2
Function: Suppresses apoptosis in a variety of cell systems including factor-dependent lymphohematopoietic and neural cells. Regulates cell death by controlling the mitochondrial membrane permeability. Appears to function in a feedback loop system with caspases. Inhibits caspase activity either by preventing the release of cytochrome c from the mitochondria and/or by binding to the apoptosis-activating factor (APAF-1). May attenuate inflammation by impairing NLRP1-inflammasome activation, hence CASP1 activation and IL1B release.

Gene Name: BCL2L1
Protein: Bcl-2-like protein 1
Function: Potent inhibitor of cell death. Inhibits activation of caspases. Appears to regulate cell death by blocking the voltage-dependent anion channel (VDAC) by binding to it and preventing the release of the caspase activator, CYC1, from the mitochondrial membrane. Also acts as a regulator of G2 checkpoint and progression to cytokinesis during mitosis. Isoform Bcl-X(L) also regulates presynaptic plasticity, including neurotransmitter release and recovery, number of axonal mitochondria as well as size and number of synaptic vesicle clusters. During synaptic stimulation, increases ATP availability from mitochondria through regulation of mitochondrial membrane ATP synthase F1F0 activity and regulates endocytic vesicle retrieval in hippocampal neurons through association with DMN1L and stimulation of its GTPase activity

in synaptic vesicles. May attenuate inflammation impairing NLRP1-inflammasome activation, hence CASP1 activation and IL1B release.

Gene Name: CCL2
Protein: C-C motif chemokine 2
Function: Chemotactic factor that attracts monocytes and basophils but not neutrophils or eosinophils. Augments monocyte anti-tumor activity. Has been implicated in the pathogenesis of diseases characterized by monocytic infiltrates, like psoriasis, rheumatoid arthritis or atherosclerosis. May be involved in the recruitment of monocytes into the arterial wall during the disease process of atherosclerosis.

Gene Name: JAG1
Protein: Protein jagged-1
Function: Ligand for multiple Notch receptors and involved in the mediation of Notch signaling. May be involved in cell-fate decisions during hematopoiesis. Seems to be involved in early and late stages of mammalian cardiovascular development. Inhibits myoblast differentiation (By similarity). Enhances fibroblast growth factor-induced angiogenesis (in vitro).

*Gene Name:*BMP5 *Protein:* Bone morphogenetic protein 5 *Function:* Induces cartilage and bone formation.

Gene Name: VEGFC
Protein: Vascular endothelial growth factor C
Function: Growth factor active in angiogenesis, and endothelial cell growth, stimulating their proliferation and migration and also has effects on the permeability of blood vessels. May function in angiogenesis of the venous and lymphatic vascular systems during embryogenesis, and also in the maintenance of differentiated lymphatic endothelium in adults. Binds and activates VEGFR-2 (KDR/FLK1) and VEGFR-3 (FLT4) receptors.

Gene Name: TNFRSF1A
Protein: Tumor necrosis factor receptor superfamily member 1A
Function: Receptor for TNFSF2/TNF-alpha and homotrimeric TNFSF1/lymphotoxin-alpha. The adapter molecule FADD recruits caspase-8 to the activated receptor. The resulting death-inducing signaling complex (DISC) performs caspase-8 proteolytic activation which initiates the subsequent cascade of caspases (aspartate-specific cysteine proteases) mediating apoptosis. Contributes to the induction of non-cytocidal TNF effects including anti-viral state and activation of the acid sphingomyelinase.

Gene Name: STAT1 *Protein:* Signal transducer and activator of transcription 1-alpha/beta
Function: Signal transducer and transcription activator that mediates cellular responses to interferons (IFNs), cytokine KITLG/SCF and other cytokines and other growth factors. Following type I IFN (IFN-alpha and IFN-beta) binding to cell surface receptors, signaling via protein kinases leads to activation of Jak kinases (TYK2 and JAK1) and to tyrosine phosphorylation of STAT1 and STAT2. The phosphorylated STATs dimerize and associate with ISGF3G/IRF-9 to form a complex termed ISGF3 transcription factor, that enters the nucleus. ISGF3 binds to the IFN stimulated response element (ISRE) to activate the transcription of IFN-stimulated genes (ISG), which drive the cell in an antiviral state. In response to type II IFN (IFN-gamma), STAT1 is tyrosine- and serine-phosphorylated. It then forms a homodimer termed IFN-gamma-activated factor (GAF), migrates into the nucleus and binds to the IFN gamma activated sequence (GAS) to drive the expression of the target genes, inducing a cellular antiviral state. Becomes activated in response to KITLG/SCF and KIT signaling. May mediate cellular responses to activated FGFR1, FGFR2, FGFR3 and FGFR4.

Gene Name: TIMP2
Protein: Metalloproteinase inhibitor 2
Function: Complexes with metalloproteinases (such as collagenases) and irreversibly inactivates them by binding to their catalytic zinc cofactor. Known to act on MMP-1, MMP-2, MMP-3, MMP-7, MMP-8, MMP-9, MMP-10, MMP-13, MMP-14, MMP-15, MMP-16 and MMP-19.

Gene Name: LTB4R
Protein: Leukotriene B4 receptor 1
Function: Receptor for extracellular ATP > UTP and ADP. The activity of this receptor is mediated by G proteins which activate a phosphatidylinositol-calcium second messenger system. May be the cardiac P2Y receptor involved in the regulation of cardiac muscle contraction through modulation of L-type calcium currents. Is a receptor for leukotriene B4, a potent chemoattractant involved in inflammation and immune response.

Gene Name: GDF5
Protein: Growth/differentiation factor 5
Function: Growth factor involved in bone and cartilage formation. During cartilage development regulates differentiation of chondrogenic tissue through two pathways. Firstly, positively regulates differentiation of chondrogenic tissue through its binding of high affinity with BMPR1B and of less affinity with BMPR1A, leading to induction of SMAD1-SMAD5-SMAD8 complex phosphorylation and then SMAD protein signaling transduction (PubMed:24098149, PubMed:21976273, PubMed:15530414, PubMed:25092592). Secondly, negatively regulates chondrogenic differentiation through its interaction with NOG (PubMed:21976273). Required to prevent excessive muscle loss upon denervation. This function requires SMAD4 and is mediated by phosphorylated SMAD1/5/8 (By similarity). Binds bacterial lipopolysaccharide (LPS) and mediates LPS-induced inflammatory response, including TNF secretion by monocytes.

Gene Name: DLL3
Protein: Delta-like protein 3
Function: Inhibits primary neurogenesis. May be required to divert neurons along a specific differentiation pathway. Plays a role in the formation of somite boundaries during segmentation of the paraxial mesoderm (By similarity).

Gene Name: BCL3
Protein: B-cell lymphoma 3 protein
Function: Contributes to the regulation of transcriptional activation of NF-kappa-B target genes. In the cytoplasm, inhibits the nuclear translocation of the NF-kappa-B p50 subunit. In the nucleus, acts as transcriptional activator that promotes transcription of NF-kappa-B target genes. Contributes to the regulation of cell proliferation (By similarity).

Gene Name: IGFBP3
Protein: Insulin-like growth factor-binding protein 3
Function: IGF-binding proteins prolong the half-life of the IGFs and have been shown to either inhibit or stimulate the growth promoting effects of the IGFs on cell culture. They alter the interaction of IGFs with their cell surface receptors. Also exhibits IGF-independent antiproliferative and apoptotic effects mediated by its receptor TMEM219/IGFBP-3R.

Gene Name: CX3CL1
Protein: Fractalkine
Function: Acts as a ligand for both CX3CL1 and integrins. Binds to CX3CR1 (PubMed:23125415, PubMed:9931005, PubMed:21829356). Binds to integrins ITGAV:ITGB3 and ITGA4:ITGB1.

Can activate integrins in both a CX3CR1-dependent and CX3CR1-independent manner. In the presence of CX3CR1, activates integrins by binding to the classical ligand-binding site (site 1) in integrins. In the absence of CX3CR1, binds to a second site (site 2) in integrins which is distinct from site 1 and enhances the binding of other integrin ligands to site 1 (PubMed:23125415, PubMed:24789099). The soluble form is chemotactic for T-cells and monocytes and not for neutrophils. The membrane-bound form promotes adhesion of those leukocytes to endothelial cells. May play a role in regulating leukocyte adhesion and migration processes at the endothelium.

Gene Name: CXCL6
Protein: C-X-C motif chemokine 6
Function: Chemotactic for neutrophil granulocytes. Signals through binding and activation of its receptors (CXCR1 and CXCR2). In addition to its chemotactic and angiogenic properties, it has strong antibacterial activity against Gram-positive and Gram-negative bacteria (90-fold-higher when compared to CXCL5 and CXCL7).

Gene Name: VEGFA
Protein: Vascular endothelial growth factor A
Function: Growth factor active in angiogenesis, vasculogenesis and endothelial cell growth. Induces endothelial cell proliferation, promotes cell migration, inhibits apoptosis and induces permeabilization of blood vessels. Binds to the FLT1/VEGFR1 and KDR/VEGFR2 receptors, heparan sulfate and heparin. NRP1/Neuropilin-1 binds isoforms VEGF-165 and VEGF-145. Isoform VEGF165B binds to KDR but does not activate downstream signaling pathways, does not activate angiogenesis and inhibits tumor growth.

Gene Name: SHC1
Protein: SHC-transforming protein 1
Function: Signaling adapter that couples activated growth factor receptors to signaling pathways. Participates in a signaling cascade initiated by activated KIT and KITLG/SCF. Isoform p46Shc and isoform p52Shc, once phosphorylated, couple activated receptor tyrosine kinases to Ras via the recruitment of the GRB2/SOS complex and are implicated in the cytoplasmic propagation of mitogenic signals. Isoform p46Shc and isoform p52Shc may thus function as initiators of the Ras signaling cascade in various non-neuronal systems. Isoform p66Shc does not mediate Ras activation, but is involved in signal transduction pathways that regulate the cellular response to oxidative stress and life span. Isoform p66Shc acts as a downstream target of the tumor suppressor p53 and is indispensable for the ability of stress-activated p53 to induce elevation of intracellular oxidants, cytochrome c release and apoptosis. The expression of isoform p66Shc has been correlated with life span (By similarity). Participates in signaling downstream of the angiopoietin receptor TEK/TIE2, and plays a role in the regulation of endothelial cell migration and sprouting angiogenesis.

Bibliography

Cancer Research UK Kidney Cancer Statistics, 2015. URL http://www.cancerresearchuk.org/cancer-info/cancerstats/types/kidney/.

BCL2L1 Genecard. http://www.genecards.org/cgi-bin/carddisp.pl?gene=BCL2L1, 2016. [Online; accessed 25-August-2016].

Uniprot CD276 Human Gene. http://www.uniprot.org/uniprot/Q5ZPR3, 2016. [Online; accessed 25-August-2016].

NIH Overview of Angiogenesis. http://www.ncbi.nlm.nih.gov/books/NBK53238/, 2016. [Online; accessed 07-August-2016].

Uniprot BCL2 Gene. http://www.uniprot.org/uniprot/P10415, 2016. [Online; accessed 25-August-2016].

CAKI1 Human Cell Line. https://www.mskcc.org/research-advantage/support/technology/tangible-material/caki-1-human-renal-cell-line, 2016. [Online; accessed 16-August-2016].

Anaesthetist Statistical Calculations. http://www.anaesthetist.com/mnm/stats/roc/Findex.htm, 2016. [Online; accessed 12-August-2016].

Cancer Gov NCI Dictionary of Cancer Terms. http://www.cancer.gov/publications/dictionaries/cancer-terms?cdrid=561720, 2016a. [Online; accessed 06-Aug-2016].

Cancer Gov Understanding Cancer. http://www.cancer.gov/about-cancer/understanding/what-is-cancer, 2016b. [Online; accessed 07-August-2016].

Cancer Hallmarks Summary. http://www.jargonwall.com/cancer/introduction-hallmarks-cancer/, 2016c. [Online; accessed 07-August-2016].

Cell Differentiation. http://study.com/academy/lesson/what-is-cell-differentiation-process-importance-examples.html, 2016a. [Online; accessed 06-Aug-2016].

Cell Line Terminology. http://medical-dictionary.thefreedictionary.com/cellline, 2016b. [Online; accessed 30-August-2016].

Cancer Gov c-KIT. http://www.cancer.gov/publications/dictionaries/cancer-terms?cdrid=44329, 2016. [Online; accessed 16-August-2016].

NIH DNA. https://ghr.nlm.nih.gov/primer/basics/dna, 2016. [Online; accessed 22-July-2016].

EMBL-EBI Protein. http://www.ebi.ac.uk, 2016. [Online; accessed 22-July-2016].

Gene Expression Analysis. http://www.bio-rad.com/en-uk/applications-technologies/what-gene-expression-analysis, 2016a. [Online; accessed 14-August-2016].

Nature gene regulation. http://www.nature.com/scitable/topicpage/gene-expression-14121669, 2016b. [Online; accessed 06-Aug-2016].

Histology. http://www.mesothelioma-aid.org/histology.htm, 2016. [Online; accessed 26-July-2016].

Reverse Engineering and Identification in Systems Biology. http://rsif.royalsocietypublishing.org/content/11/91/20130505, 2016. [Online; accessed 28-August-2016].

Igraph. http://igraph.org/, 2016. [Online; accessed 15-July-2016].

NIH Cancer genome. http://cancergenome.nih.gov/cancersselected/kidneyclearcell, 2016. [Online; accessed 16-August-2016].

MRC Laboratory of Molecular Biology Microarray. http://www.mrc-lmb.cam.ac.uk/genomes/madanm/microarray/chapter-final.pdf, 2016a. [Online; accessed 13-July-2016].

Microarray Data Analysis. http://www.ub.edu/stat/docencia/bioinformatica/microarrays/ADM/slides/A_Tutorial_Review_of_Microarray_data_Analysis_17-06-08.pdf, 2016b. [Online; accessed 13-July-2016].

NIH Protein. http://www.ncbi.nlm.nih.gov, 2016. [Online; accessed 22-July-2016].

NHS Cancer Introduction. http://www.nhs.uk/conditions/cancer/Pages/Introduction.aspx, 2016. [Online; accessed 13-July-2016].

Uniprot PDGFR. http://www.uniprot.org/uniprot/P09619, 2016. [Online; accessed 16-August-2016].

EasyCalculation P-value Calculation. https://www.easycalculation.com/statistics/p-value-for-z-score.php, 2016. [Online; accessed 12-August-2016].

Cancer Gov RCC Treatment. http://www.cancer.gov/types/kidney/patient/kidney-treatment-pdq, 2016. [Online; accessed 15-August-2016].

R Language. https://www.r-project.org/, 2016. [Online; accessed 26-July-2016].

rTMA. https://github.com/alubbock/rTMA, 2016. [Online; accessed 15-July-2016].

Introduction to Stationary Distributions. http://www.mast.queensu.ca/~stat455/lecturenotes/set3.pdf, 2016. [Online; accessed 07-July-2016].

Online Statistics Education, Standard Error. http://onlinestatbook.com/2/regression/accuracy.html, 2016. [Online; accessed 12-August-2016].

EMBL-EBI VEGFR1 Interpro. http://www.ebi.ac.uk/interpro/entry/IPR009135, 2016. [Online; accessed 16-August-2016].

Nature Western Blot. http://www.nature.com/scitable/definition/western-blot-288, 2016. [Online; accessed 13-August-2016].

WHO Media Centre Cancer Statistics. http://www.who.int/mediacentre/factsheets/fs297/en/, 2016. [Online; accessed 15-August-2016].

Ziv Bar-Joseph, Anthony Gitter, and Itamar Simon. Studying and Modelling Dynamic Biological Processes using Time-series Gene Expression Data. *Nature Reviews Genetics*, 13(8):552–564, 2012.

Matthew J Beal, Zoubin Ghahramani, and Carl E Rasmussen. The Infinite Hidden Markov Model. In *Advances in neural information processing systems*, pages 577–584, 2001.

Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov Chain Monte Carlo*. CRC press, 2011.

Peter Walter Julian Lewis Martin Raff Keith Roberts Bruce Alberts, Alexander Johnson. *Molecular Biology of the Cell*. Garland Science, New York, 2002.

Ronald Bukowski and Andrew C Novick. *Renal Cell Carcinoma: Molecular Targets and Clinical Applications*. Springer, 3 edition, 2015.

K-H Cho, S-M Choo, SH Jung, J-R Kim, H-S Choi, and J Kim. Reverse Engineering of Gene Regulatory Networks. *Systems Biology*, 1(3):149–163, 2007.

Francis Crick et al. Central Dogma of Molecular Biology. *Nature*, 227(5258):561–563, 1970.

Patrik DâĂŹhaeseleer, Shoudan Liang, and Roland Somogyi. Genetic Network Inference: From Co-expression Clustering to Reverse Engineering. *Bioinformatics*, 16 (8):707–726, 2000.

Sean R Eddy. What is a Hidden Markov Model? *Nature Biotechnology*, 22(10): 1315–1316, 2004.

Isaiah J Fidler. The Pathogenesis of Cancer Metastasis: The 'Seed and Soil' Hypothesis Revisited. *Nature Reviews Cancer*, 3(6):453–458, 2003.

Valdimir Filkov. Identifying Gene Regulatory Networks from Gene Expression Data. *Handbook of Computational Molecular Biology*, pages 27–1, 2005.

Michael Finegold and Mathias Drton. Robust Graphical Modeling of Gene Networks using Classical and Alternative T-distributions. *The Annals of Applied Statistics*, pages 1057–1080, 2011.

Emily B Fox, Erik B Sudderth, Michael I Jordan, and Alan S Willsky. An HDP-HMM for Systems with State Persistence. In *Proceedings of the 25th International Conference on Machine learning*, pages 312–319. ACM, 2008.

Emily B Fox, Erik B Sudderth, Michael I Jordan, and Alan S Willsky. A Sticky HDP-HMM with Application to Speaker Diarization. *The Annals of Applied Statistics*, pages 1020–1056, 2011.

Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian Network Classifiers. *Machine Learning*, 29(2-3):131–163, 1997.

Nir Friedman, Michal Linial, Iftach Nachman, and Dana Pe'er. Using Bayesian Networks to Analyze Expression Data. *Journal of Computational Biology*, 7(3-4):601–620, 2000.

Matthew D Galsky. A Prognostic Model for Metastatic Renal Cell Carcinoma. *The Lancet Oncology*, 14(2):102–103, 2013.

Zoubin Ghahramani. An Introduction to Hidden Markov Models and Bayesian Networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 15 (01):9–42, 2001.

Alex Greenfield, Aviv Madar, Harry Ostrer, and Richard Bonneau. DREAM4: Combining Genetic and Dynamic Information to Identify Biological Networks and Dynamical Models. *PloS one*, 5(10):e13397, 2010.

AE Greijer and E Van der Wall. The Role of Hypoxia Inducible Factor 1 (HIF-1) in Hypoxia Induced Apoptosis. *Journal of Clinical Pathology*, 57(10):1009–1014, 2004.

Marco Grzegorczyk, Dirk Husmeier, Kieron D Edwards, Peter Ghazal, and Andrew J Millar. Modelling Non-stationary Gene Regulatory Processes with a Non-homogeneous Bayesian Network and the Allocation Sampler. *Bioinformatics*, 24 (18):2071–2078, 2008.

Hendrik Hache, Hans Lehrach, and Ralf Herwig. Reverse Engineering of Gene Regulatory Networks: A Comparative Study. *EURASIP Journal on Bioinformatics and Systems Biology*, 2009:8, 2009.

Douglas Hanahan and Robert A Weinberg. The Hallmarks of Cancer. *cell*, 100(1): 57–70, 2000.

Douglas Hanahan and Robert A Weinberg. Hallmarks of Cancer: The Next Generation. *Cell*, 144(5):646–674, 2011.

Alexander J Hartemink, David K Gifford, Tommi S Jaakkola, Richard A Young, et al. Using Graphical Models and Genomic Expression Data to Statistically Validate Models of Genetic Regulatory Networks. In *Pacific Symposium on Biocomputing*, volume 6, page 266, 2001.

Seiya Imoto, Sunyong Kim, Takao Goto, Sachiyo Aburatani, Kousuke Tashiro, Satoru Kuhara, and Satoru Miyano. Bayesian Network and Nonparametric Heteroscedastic Regression for Nonlinear Modeling of Genetic Network. *Journal of Bioinformatics and Computational Biology*, 1(02):231–252, 2003.

Nazar Jawhar. Tissue Microarray: A Rapidly Evolving Diagnostic and Research Tool. *Annals of Saudi Medicine*, 29(2):123, 2009.

W Evan Johnson, Cheng Li, and Ariel Rabinovic. Adjusting Batch Effects in Microarray Expression Data using Empirical Bayes Methods. *Biostatistics*, 8(1):118–127, 2007.

Harri Lähdesmäki, Ilya Shmulevich, and Olli Yli-Harja. On Learning Gene Regulatory Networks under the Boolean Network Model. *Machine Learning*, 52(1-2):147–167, 2003.

Sophie Lebre, Jennifer Becq, Frederic Devaux, Michael PH Stumpf, and Gaelle Lelandais. Statistical Inference of the Time-varying Structure of Gene Regulation Networks. *Systems Biology*, 4(1):130, 2010.

Shoudan Liang, Stefanie Fuhrman, and Roland Somogyi. Reveal: A General Reverse Engineering Algorithm for Inference of Genetic Network Architectures. 1998.

Alex Lubbock. rTMA: A Tissue Microarray Package for R, 2016. URL `rTMA-Vignette.pdf`. [Pdf; accessed 26-July-2016].

John Ludbrook. Should We Use One-sided or Two-sided P-values in Tests of Significance? *Clinical and Experimental Pharmacology and Physiology*, 40(6):357–361, 2013.

Ole Lund. *Immunological Bioinformatics*. MIT Press, 2005.

Shisong Ma, Qingqiu Gong, and Hans J Bohnert. An Arabidopsis Gene Network Based on the Graphical Gaussian Model. *Genome Research*, 17(11):1614–1625, 2007.

Xiaomei Ma and Herbert Yu. Global Burden of Cancer. *Yale J Biol Med*, 79(3-4): 85–94, 2006.

Alberto Mantovani, Paola Allavena, Antonio Sica, and Frances Balkwill. Cancer-related Inflammation. *Nature*, 454(7203):436–444, 2008.

SR McKeown. Defining Normoxia, Physoxia and Hypoxia in Tumours - Implications for Treatment Response. *The British journal of radiology*, 87(1035):20130676, 2014.

Robert J Motzer, Neil H Bander, and David M Nanus. Renal Cell Carcinoma. *New England Journal of Medicine*, 335(12):865–875, 1996.

Robert J Motzer, Brian I Rini, Ronald M Bukowski, Brendan D Curti, Daniel J George, Gary R Hudes, Bruce G Redman, Kim A Margolin, Jaime R Merchan, George Wilding, et al. Sunitinib in Patients with Metastatic Renal Cell Carcinoma. *Jama*, 295 (21):2516–2524, 2006.

Kevin Murphy, Saira Mian, et al. Modelling Gene Expression Data using Dynamic Bayesian Networks. Technical report, Technical report, Computer Science Division, University of California, Berkeley, CA, 1999.

Natsu Nakajima and Tatsuya Akutsu. Exact and Heuristic Methods for Network Completion for Time-varying Genetic Networks. *BioMed Research International*, 2014.

Hideki Niimi, Katerina Pardali, Michael Vanlandewijck, Carl-Henrik Heldin, and Aristidis Moustakas. Notch Signaling is Necessary for Epithelial Growth Arrest by TGF-β. *The Journal of Cell Biology*, 176(5):695–707, 2007.

Naoyo Nishida, Hirohisa Yano, Takashi Nishida, Toshiharu Kamura, and Masamichi Kojiro. Angiogenesis in Cancer. *Vascular Health and Risk Management*, 2(3):213, 2006.

Melissa J Parsons and Douglas R Green. Mitochondria in Cell Death. *Essays in Biochemistry*, 47:99–114, 2010.

Christopher A Penfold and David L Wild. How to Infer Gene Networks from Expression Profiles, Revisited. *Interface Focus*, 1(6):857–870, 2011.

Christopher A Penfold, Vicky Buchanan-Wollaston, Katherine J Denby, and David L Wild. Nonparametric Bayesian Inference for Perturbed and Orthologous Gene Regulatory Networks. *Bioinformatics*, 28(12):i233–i241, 2012.

Lawrence Rabiner and B Juang. An Introduction to Hidden Markov Models. *ASSP Magazine*, 3(1):4–16, 1986.

Nader Rahimi. VEGFR-1 and VEGFR-2: Two Non-identical Twins with a Unique Physiognomy. *Frontiers in Bioscience: A journal and Virtual Library*, 11:818, 2006.

Andrew G Renehan, Catherine Booth, and Christopher S Potten. What is Apoptosis, and Why is it Important? *British Medical Journal*, 322(7301):1536, 2001.

Juliane Schäfer and Korbinian Strimmer. An Empirical Bayes Approach to Inferring Large-scale Gene Association Networks. *Bioinformatics*, 21(6):754–764, 2005.

Juliane Schäfer, Rainer Opgen-Rhein, and Korbinian Strimmer. Reverse Engineering Genetic Networks using the GeneNet Package. *J Am Stat Assoc*, 96:1151–1160, 2001.

E Schömig and CL Schönfeld. Extraneuronal Noradrenaline Transport (uptake2) in a Human Cell Line (Caki-1 Cells). *Naunyn-Schmiedeberg's Archives of Pharmacology*, 341(5):404–410, 1990.

Le Song, Mladen Kolar, and Eric P Xing. Time-varying Dynamic Bayesian Networks. In *Advances in Neural Information Processing Systems*, pages 1732–1740, 2009.

Jai-Yoon Sul, K Wu Chia-wen, Fanyi Zeng, Jeanine Jochems, Miler T Lee, Tae Kyung Kim, Tiina Peritz, Peter Buckley, David J Cappelleri, Margaret Maronski, et al. Transcriptome Transfer Produces a Predictable Cellular Phenotype. *Proceedings of the National Academy of Sciences*, 106(18):7624–7629, 2009.

Yee Whye Teh. Dirichlet process. In *Encyclopedia of machine learning*, pages 280–287. Springer, 2011.

Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical Dirichlet Processes. *Journal of the American Statistical Association*, 2012.

Thomas Thorne and Michael PH Stumpf. Inference of Temporally Varying Bayesian Networks. *Bioinformatics*, 28(24):3298–3305, 2012.

Michael J Thun, John Oliver DeLancey, Melissa M Center, Ahmedin Jemal, and Elizabeth M Ward. The Global Burden of Cancer: Priorities for Prevention. *Carcinogenesis*, 31(1):100–110, 2010.

Sergio Vázquez, Luis León, Ovidio Fernández, Martín Lázaro, Enrique Grande, and Luis Aparicio. Sunitinib: The First to Arrive at First-line Metastatic Renal Cell Carcinoma. *Advances in Therapy*, 29(3):202–217, 2012.

Mauno Vihinen. How to Evaluate Performance of Prediction Methods? Measures and their Interpretation in Variation Effect Analysis. *BMC Genomics*, 13(4):1, 2012.

Ting Wang, Zhao Ren, Ying Ding, Zhou Fang, Zhe Sun, Matthew L MacDonald, Robert A Sweet, Jieru Wang, and Wei Chen. FastGGM: An Efficient Algorithm for the Inference of Gaussian Graphical Model in Biological Networks. *PLoS Comput Biol*, 12(2):e1004755, 2016.

Zhong Wang, Mark Gerstein, and Michael Snyder. RNA-Seq: A Revolutionary Tool for Transcriptomics. *Nature Reviews Genetics*, 10(1):57–63, 2009.

Adriano V Werhli and Dirk Husmeier. Gene Regulatory Network Reconstruction by Bayesian Integration of Prior Knowledge and/or Different Experimental Conditions. *Journal of Bioinformatics and Computational Biology*, 6(03):543–572, 2008.

Jiri Zavadil, Lukas Cermak, Noemi Soto-Nieves, and Erwin P Böttinger. Integration of TGF-β/Smad and Jagged1/Notch Signalling in Epithelial-to-Mesenchymal Transition. *The EMBO journal*, 23(5):1155–1165, 2004.

Arnold Zellner. On Assessing Prior Distributions and Bayesian Regression Analysis with G-prior Distributions. *Bayesian Inference and Decision Techniques: Essays in Honor of Bruno De Finetti*, 6:233–243, 1986.