Bayesian Inference for Cancer-related Signalling Networks

Wayne Cheung
Supervisor: Dr. Jonathan Keith
Monash University

February 2014

1 Introduction

With the development of new biological experiments that can monitor and measure thousands of genes or proteins at a time, generating huge amounts of data, efficient and robust statistical methods must be developed to discover any important relationships between particular genes or proteins when analysing the data. For example, it has been suggested recently that crosstalks, certain unwanted signalling mechanisms between particular genes, could lead to drug resistance in modern cancer therapy [1]. Uncovering such signalling networks could potentially lead to the development of more targeted and effective therapy which suppresses the crosstalks and delivers better treatment outcome.

An example of a type of biological dataset that is commonly analysed is gene expression. Gene expression datasets produced from microarray experiments provide information about the activation level of a particular gene. In a biological sample, genes which are highly activated compared to a control are said to be overexpressed, whereas those which are deactivated relative to the control are said to be underexpressed. Information from thousands of samples are collated in many databases, such as the Gene Expression Omnibus [2], many of which are publicly available. It is thought that such datasets encode information about the underlying signalling networks of the measured genes, and that upon careful analysis these networks can be uncovered.

One such method is to infer a Bayesian network based on the data collected [3]. This report will introduce the theory of Bayesian Networks, and a commonly used multinomial probability expression to infer such networks. A new Markov Chain Monte Carlo (MCMC) method, the Neighbourhood Sampler [4] is then applied to test its computational efficiency.

From simulations on small networks, it was found that the probability expression does not always recover the correct network. Nevertheless, when it does, the Neighbourhood Sampler has been shown to be an efficient way of inferring the correct network.

Due to time constraints the Neighbourhood Sampler has not been applied to real-life gene expression datasets, but this report serves to provide a good introduction to the topic, and shows some early promises based on the results from small-scale simulations.

2 Background and Methods

2.1 Bayesian Networks

2.1.1 Definitions

Bayesian networks can be represented as graphs. A simple definition of a graph G is that:

$$G = (N, E)$$

It comprises a set of nodes N and edges E.

The graphs which are used in Bayesian Networks are Directed Acyclic Graphs (DAGs), in which all the edges are directed, and no cycles are permitted.

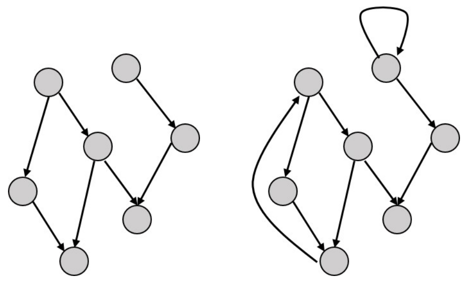


Figure 1: The graph on the left is a valid DAG, whereas the graph on the right is not because some of the directed edges form cycles.

In a Bayesian network, each of the nodes in a DAG represents a variable, $X_1, X_2, ..., X_n$, and a directed edge from X_i to X_j indicates X_j is dependent on X_i, or in other words, the value of X_i influences X_j. We call all the variables with edges pointing into a particular X_i the parents of that variable, denoted as Pa_i.

To fully describe a Bayesian network, in addition to the DAG which represents the interdependencies between the variables, each variable X_i has a conditional probability distribution dependent on its parents $P(X_i|Pa_i)$ which we denote by θ_i.

2.1.2 Markov Assumption

An important property of the Bayesian network is the Markov assumption:

$$\boxed{\text{Each variable } X_i \text{ is independent of its nondescendants, given its parents in } G.}$$

This decomposes a complicated multi-dimensional probability distribution into a product of distributions of lower dimensions.

Let's consider the joint distribution of all the variables involved in a particular network:

$$P(X_1, X_2, ..., X_n)$$

It can be factored into a series of conditional probabilities as:

$$\begin{aligned} P(X_1, X_2, ..., X_n) &= \prod_{i=1}^{n} P(X_i|X_1, X_2, ..., X_{i-1}) \\ &= P(X_1)P(X_2|X_1)P(X_3|X_1, X_2)... \\ &\quad P(X_n|X_1, X_2, ..., X_{n-1}) \end{aligned}$$

Which can then be simplified using the Markov assumption,

$$P(X_1, X_2, ...X_n) = \prod_{i=1}^{n} P(X_i|Pa_i)$$

In the example network of Figure 2, the causes and effects of a car crash are represented as directed edges from one variable to another. Looking specifically at the variable 'Car Crash', we see that it has three parent variables: 'Texting', 'Speeding' and 'Weather'. Although the variable 'Age' influences 'Texting' and 'Speeding', the Markov assumption allows us to ignore this particular dependence and focus entirely on the variable 'Car Crash' being conditionally dependent on 'Texting', 'Speeding' and 'Weather', or in a probability expression:

$$P(C|A, T, S, W) = P(C|T, S, W)$$

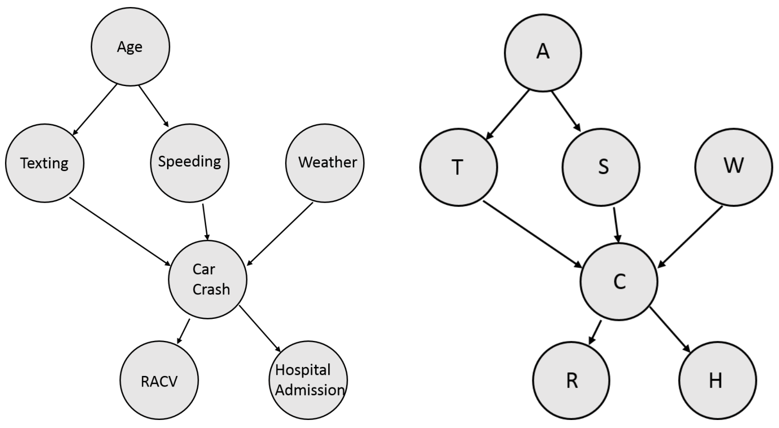


Figure 2: Example car crash network and the same network represented by initial letters.

Such conditional independencies allow us to simplify the overall probability expression:

$$\begin{aligned}
P(A,W,T,S,C,R,H) &= P(A)P(W|A)P(T|A,W)P(S|A,W,T) \\
&\quad P(C|A,W,T,S)P(R|A,W,T,S,C) \\
&\qquad P(H|A,W,T,S,C,R) \\
&= P(A)P(W)P(T|A)P(S|A)P(C|W,T,S) \\
&\quad P(I|C)P(H|C)
\end{aligned}$$

This demonstrates the compactness of representing networks as Bayesian networks.

2.1.3 Equivalent Networks

When two DAGs describe the same set of independencies between the variables, they are said to be equivalent as they are indistinguishable based on experimental observations.

To determine if two DAGs are equivalent, we first need to define a v-structure as a subgraph consisting of 3 nodes, X, Y and Z, in the form $X \to Y \leftarrow Z$ with no edges connecting X and Z.

Then two DAGs with the same number of nodes are equivalent when [5]:

1. They have the same underlying undirected graph structure, and
2. They have the same v-structures.

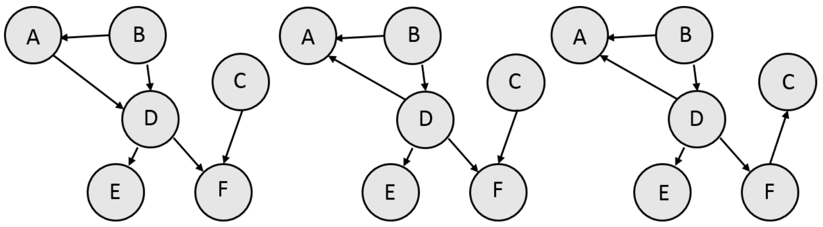


Figure 3: The leftmost graph and the middle graph are equivalent because they satisfy both criteria (the subgraph consisting of A, B and D is not a v-structure because A and B are not connected), but the middle graph and the rightmost graph are not because the v-structure $D \to F \leftarrow C$ in the middle graph has disappeared in the rightmost graph.

2.1.4 Probability Distributions

Moving away from the DAGs, we also need to consider the probability distributions $P(X_i|Pa_i)$ for each variable X_i. There are two common types of distributions for which closed form expressions of the graph probability distribution have been worked out:

1. Multinomial model:
$$P(X_i = k|Pa_i = j) = \theta_{ijk}$$
The multinomial probability that X_i has outcome k, given its parents are in the set of outcomes j, is given by θ_{ijk}.

2. Linear Gaussian model, with fixed variance σ_i^2:
$$\mu_i = \beta_{i0} + \sum_{j=1}^{p_i} \beta_{ij} x_{ij}$$
Each variable X_i is normally distributed with mean μ_i and variance σ_i^2, and the mean is a sum of a constant term β_{i0} and the value of all its parents x_{ij} multiplied by the constants β_{ij}.

In the simulations conducted in this project, only the binomial case under the multinomial model was considered as the simplest possible model to test the efficiency of the Neighbourhood Sampler MCMC technique. Though it could be easily extended to the multinomial or the linear Gaussian model.

2.1.5 Multinomial Model

There are two parts to the probability distribution of a Bayesian Network given a set of data, D, with the multinomial model:

1. We can first infer the probability distribution of the parameters θ_{ijk}, given D and G, from a Bayesian approach. Using a conjugate Dirichlet prior, the posterior distribution of the parameters is also a Dirichlet distribution, given by [6]:

$$P(\theta|D,G) = \text{Dirichlet}(\alpha_{ij1} + N_{ij1}, \alpha_{ij2} + N_{ij2}, ..., \alpha_{ijr_i} + N_{ijr_i}) \tag{1}$$

   where N_{ijk} is the number of counts of X_i having the outcome k, given its parents are in the setup j, and α_{ijk} is the corresponding imaginary counts specified in the Dirichlet prior, to indicate any prior beliefs before any measurements are made.

2. We can also construct the probability distribution of the graph G given D using Bayes' rule:

$$P(G|D) \propto P(D|G)P(G)$$

   Assuming a uniform prior on the graph $P(G)$, we have

$$P(G|D) \propto P(D|G)$$

   and we can find the likelihood $P(D|G)$ by factoring out the parameters θ and marginalising them out in the integral [6]:

$$\begin{aligned} P(D|G) &= \int P(D,\theta|G)\,d\theta \\ &= \int P(D|\theta,G)P(\theta|G)\,d\theta \\ &= \prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{\Gamma(\alpha_{ij})}{\Gamma(\alpha_{ij}+N_{ij})}\prod_{k=1}^{r_i}\frac{\Gamma(\alpha_{ijk}+N_{ijk})}{\Gamma(\alpha_{ijk})}, \qquad (2) \\ & \alpha_{ij} = \sum_{k=1}^{r_i}\alpha_{ijk}, \quad N_{ij} = \sum_{k=1}^{r_i} N_{ijk} \end{aligned}$$

Equation (2) is an important expression in Bayesian network inference using the multinomial model as it gives a probability distribution of all the possible graphs that could describe a set of data. For each of the n variables, we consider each of the q_i sets of parent values individually, and within each set of parent values, we count the number of occurrences of each outcome N_{ijk} out of the r_i outcomes for each variable. The gamma function in (2) comes from the fact that the integral involves calculating the expected value of a Dirichlet distribution.

The Dirichlet priors are assigned uniformly [7]:

$$\alpha_{ijk} = \frac{\alpha}{q_i r_i} \tag{3}$$

where α is the total imaginary counts for the Dirichlet prior, and was set to 1 in this project to represent a non-informative prior, such that:

1. $P(D|G)$ is the highest for the correct structure, and
2. Graphs which are equivalent to each other have the same probability.

2.2 Network Inference

With (3) setting up the Dirichlet priors, (2) giving the probability distribution over the graphs, and (1) giving the probability distribution over the parameters, the full probability distribution for the Bayesian networks given a set of data can be calculated. However, this is often computationally difficult because the number of DAGs increases very quickly with an increasing number of nodes. The number of DAGs for a certain number of nodes n can be evaluated iteratively using the expression [8]:

$$f(n) = \sum_{i=1}^{n} (-1)^{i+1} C_i^n 2^{i(n-i)} f(n-i)$$

As a result, algorithms have been developed to reduce the computational time required to infer the correct Bayesian network. A simple algorithm is the Greedy Search, where starting from an arbitrary graph, each time after considering all the possible edges between any two nodes that can be added without creating cycles, we add in the one which increases the probability the most. Then we keep adding in edges until the probability does not increase any more.

The Greedy Search and other similar techniques face a fundamental problem that they could be stuck at a local maximum, namely because each time only the edge

Number of variables	Number of DAGs
1	1
2	3
3	25
4	543
5	29 281
6	3 781 503
7	1 138 779 265
8	783 702 329 343
9	1 213 442 454 842 881
10	4 175 098 976 430 598 100

Table 1: Number of DAGs increases dramatically with an increasing number of variables.

which increases the probability the most is added, we might never get to the graph with the overall highest probability if in order to get there, we would first go through several graphs with lower probabilities.

2.2.1 Neighbourhood Sampler

The Neighbourhood Sampler [4] is a Markov Chain Monte Carlo (MCMC) technique which avoids the problem of being stuck at a local maximum. At each iteration it first considers all the possible edges that can be added or taken away while keeping it acyclic — all these valid DAGs plus the original graph itself are defined as its neighbourhood. However, even if none of the resultant graphs increases the graph probability, the algorithm introduces a random acceptance or rejection step which gives a slight chance of moving onto another graph such that it would not be stuck at any particular graph, but would continue to explore the whole space of DAGs.

As a sampling technique, the simulated draws should match the probability distribution of the graphs as expressed in equation (2). By being a MCMC technique, it carries out the simulation from a Markov Chain which is constructed specifically to converge to the target graph distribution. An MCMC method always converges, however this could be too slow to be worth it.

One of the main aims of the project is to use the MCMC property of the Neighbourhood Sampler to explore the space of DAGs such that it would not be stuck at any local maxima, but at the same time it would find the graph with the highest overall probability quickly enough to be computationally efficient. Instead of actually sampling the whole distribution and exploring the entire space of DAGs, it was hoped that if the probability distribution (2) is sharply peaked enough, then even without exploring the whole graph space, once the Markov Chain lands on some of the graphs with the highest probabilities, those graphs would be sampled so many more times than the others that it is obvious they are the correct networks we are trying to infer.

3 Simulations

All the simulations in this project were written in Matlab, with some running on a Dell Optiplex 9020 computer with an Intel i7-4770 (3.40 GHz) CPU and 16.0 GB RAM, and some running on the Monash Campus Cluster.

3.1 Probability Distribution over the Space of DAGs with 3 Nodes

The space of DAGs with only 3 nodes is small enough that the entire probability distribution can be evaluated. Here a problem was discovered that the graphs with the highest probability might not always be the original network from which data was simulated in the first place.

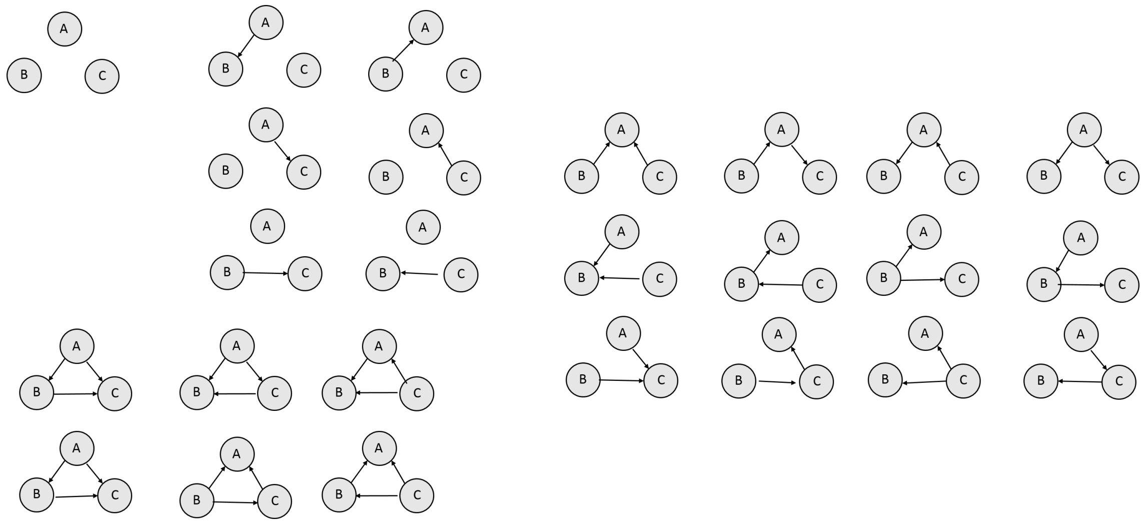


Figure 4: There are 25 DAGs with 3 nodes.

One possible explanation is that the amount of data simulated was not enough to confidently narrow it down to the correct network, as additional randomness would have been introduced in the simulation process.

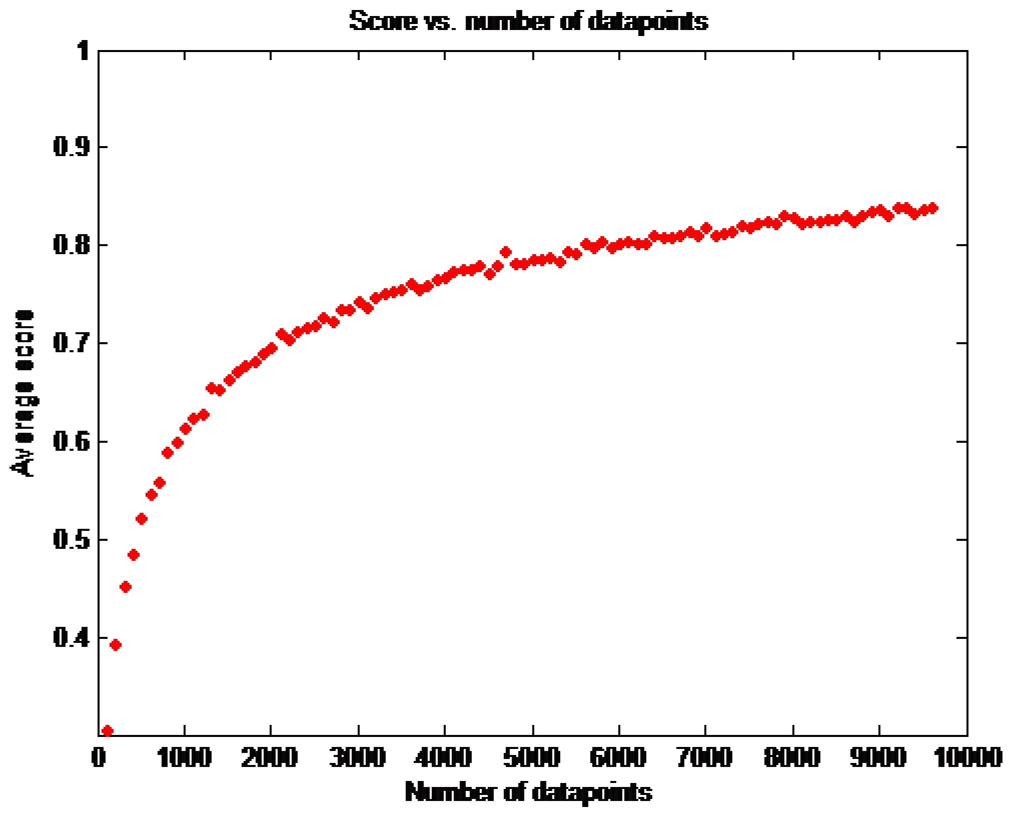


Figure 5: In each of the 10 000 runs, the probability of the original structure was recorded, and then averaged over the 10 000 runs to see if increasing the amount of data generated would increase the probability of recovering the original structure.

Figure 5 shows as more datapoints are simulated, the original structure is more likely to be recovered, indicating this could be one of the contributing factors. However, in real-life experiments, it is unlikely that more than 10 000 datapoints could be measured. Realistically, the number of repeated measurements would fall into the hundreds, or maybe even just below a hundred, such that graph with the highest probability given by expression (2) might not be the correct graph.

A potential method to get around this is a bootstrap method [3] where original datasets are scrambled to give more data that could be analysed. The validity of this method might be questionable, however, and requires further testing and mathematical work.

3.2 Testing the Neighbourhood Sampler

5000 datapoints were simulated when the Neighbourhood Sampler was tested on the 3-node networks. As the number of iterations increases, the sampled draws better match the true distribution, as indicated by the series of graphs in Figure 6 and a decreasing sum of squares of differences between the two distributions as shown in Figure 7. This demonstrates good convergence. Figure 8 shows only a small degree of burn-in is required.

The simulation with 3-node DAGs shows the Neighbourhood Sampler achieves good convergence relatively quickly, which is a promising start, even if the probability expression does not behave as expected to recover the original structure.

This was then extended to a larger network consisting of 7 nodes. 10 000 datapoints were simulated, and 5000 iterations of the Neighbourhood Sampler were run. Before considering any burn-in, Figure 9 already shows all the equivalent graphs with the same true probability are sampled significantly more than all the other graphs, suggesting the Neighbourhood Sampler has quite successfully reached the high probability density region, and stays there for long enough to indicate so. It could also be checked that these graphs with the highest probabilities are in the same equivalent class to the original network, so this was a successful simulation where the original structure was recovered.

Looking at the log likelihood graph in Figure 10, it seems reasonable to discard the first 600 iterations as burn-in.

Doing so we obtain Figure 11, and we see once the burn-in period is introduced, most of the sampled graphs fall inside the high probability density region, indicating the Neighbourhood Sampler has sampled the high probability density region enough to recover the correct original network.

4 Conclusions

From the simulations, it can be seen that the Neighbourhood Sampler has been quite effective at exploring the space of DAGs and reaching the high probability density region. This is a promising start to use this technique to infer Bayesian networks given a set of data.

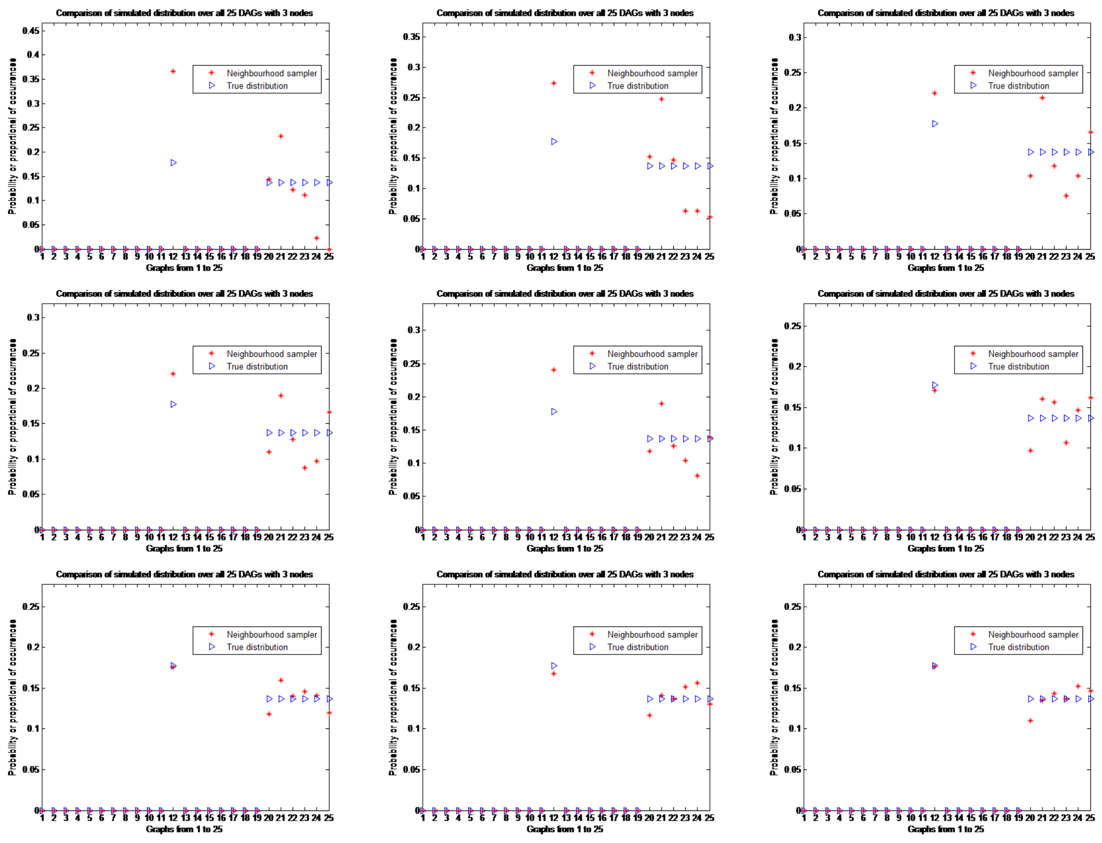
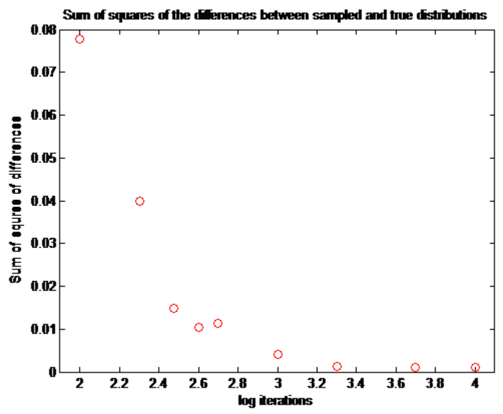


Figure 6: From left to right, top to bottom, the graphs compare the sampled distribution and the true probability with 100, 200, 300, 400, 500, 1000, 2000, 5000 and 10 000 iterations respectively. The original structure was graph 21, to which graphs 20 to 25 are equivalent. Note that the sampled distribution still resembles the true distribution, even if the graph with the highest probability is not the original structure where data was simulated from.

The main barrier still to be overcome is to understand the probability expression better and if there is a way to get around the fact that the graphs with the highest probability might not necessarily be equivalent to the correct network.

Then, once further testing has been done to test the algorithm's robustness and rate of convergence, it could be applied onto real-life gene expression data to discover the underlying signalling networks between genes. To speed up the algorithm it may also be reasonable to restrict the space of DAGs from which the sampling is taking place by



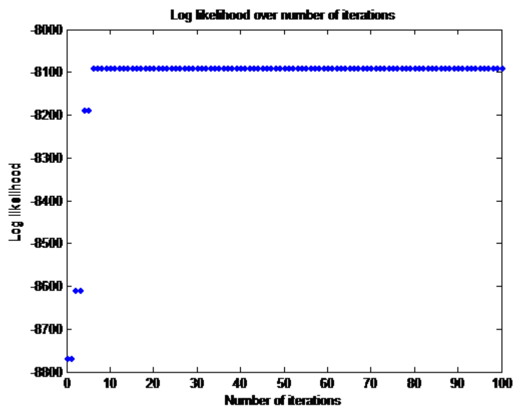


Figure 7: Graph on the left plots the sum of squares of differences between the sampled and true distributions against log iterations, and shows this difference diminishes with an increasing number of iterations, again demonstrating convergence at a resonable speed.

Figure 8: Graph on the right is a plot of the log likelihood at each iteration for 100 iterations. It shows convergence is quick and based on this graph a 10-iteration burn-in was introduced to all of the 3-node simulations.

restricting the connectivity, for example by requiring all the nodes to be connected to at least one other node, or setting a maximum number of parent nodes for each node.

A final note which should be mentioned is that because of equivalence, it is often not possible to pin it down a specific graph given a set of data. More often it is only possible to find a whole class of equivalent graphs which have the same probability and are therefore indistinguishable. To determine the exact network out of the equivalent graphs, intervention is necessary [5] to deliberately set the values of some of the variables, and determine the correct network based on the effects on the other variables.

5 Acknowledgement

I would like to thank AMSI and CSIRO for the vacation scholarship and the wonderful opportunity to participate in the Big Day In to not only give my presentation, but also to listen to the amazing work that other vacation research scholars have done. It has been a worthwhile experience learning about this topic and running the simulations

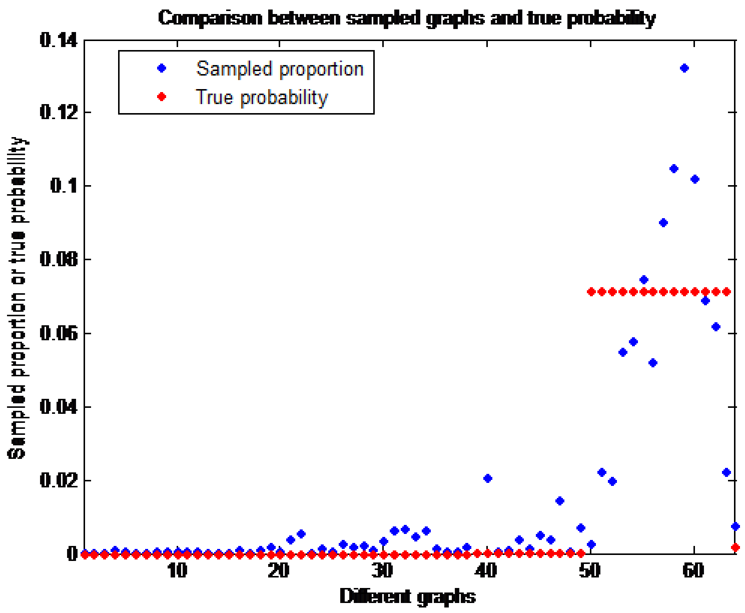


Figure 9: 5000 iterations of the Neighbourhood Sampler on the 7-node network.

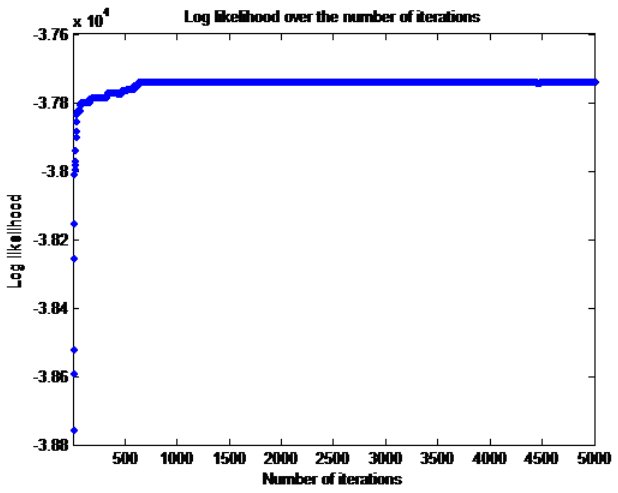


Figure 10: Log likelihood graph of 5000 iterations of the Neighbourhood Sampler.

to test my ideas out. I also like to thank Monash University for providing all of us vacation scholars with a nice room and some rather powerful computers for us to work

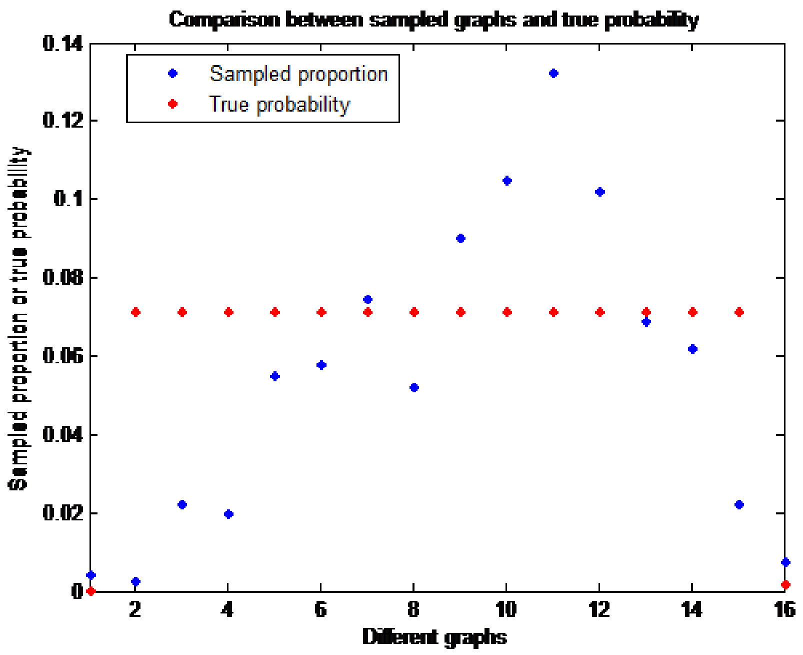


Figure 11: Introducing a 600-iteration burn-in on the 7-node network.

with. And of course I should also need to thank my supervisor, Dr Jonathan Keith, for taking me through the project, providing guidance and helpful hints all along the way and pushing me back onto the right track from time to time.

References

[1] Yamaguchi, H, Chang, S-S, Hsu, JL & Hung, M-C 2014, ‘Signaling cross-talk in the resistance to HER family receptor targeted therapy’, *Oncogene*, issue 33, pp. 1073-1081.

[2] National Center for Biotechnology Information 2014, *Gene Expression Omnibus*, US National Library of Medicine, Bethesda, USA, viewed 27 Feb 2014, http://www.ncbi.nlm.nih.gov/geo/

[3] Friedman, N, Linial, M, Nachman, I & Pe’e, D 2000, ‘Using Bayesian Networks to Analyze Expression Data’, *Journal of Computational Biology*, vol. 7, no. 3/4, pp. 601-620.

[4] Keith, J, Sofronov, G & Kroese, D 2008, 'The Generalized Gibbs Sampler and the Neighborhood Sampler', in Keller, A, Heinrich, S & Niederreiter, H (Eds.), *Monte Carlo and Quasi-Monte Carlo Methods 2006*, Springer Berlin Heidelberg.

[5] Pe'er, D 2005, 'Bayesian Network Analysis of Signaling Networks: A Primer', *Science Signaling*, issue 281, p. pl4.

[6] Heckerman, D 1995, *A Tutorial on Learning With Bayesian Networks*, Microsoft Research, Redmond.

[7] Buntine, W 1991, 'Theory Refinement on Bayesian Networks', *Proceedings of the Seventh Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-91)*, Morgan Kaufmann, San Mateo, California, pp. 52-60.

[8] Robinson, R, 'Counting Labeled Acyclic Digraphs', in Harary, F (Ed.), *New Directions in the Theory of Graphs*, Academic Press, New York, pp. 239-273.