Inference and Decision Making in Large Weakly Dependent Graphical Models

Lisa Jane Turner, M.Math (Hons.), M.Res

Submitted for the degree of Doctor of Philosophy at Lancaster University.

April 2017

Abstract

This thesis considers the problem of searching a large set of items, such as emails, for a small subset which are relevant to a given query. This can be implemented in a sequential manner – whereby knowledge from items that have already been screened is used to assist in the selection of subsequent items to screen. Often the items being searched have an underlying network structure. Using the network structure and a modelling assumption that relevant items and participants are likely to cluster together can greatly increase the rate of screening relevant items. However, inference in this type of model is computationally expensive.

In the first part of this thesis, we show that Bayes linear methods provide a natural approach to modelling this data. We develop a new optimisation problem for Bernoulli random variables, called constrained Bayes linear, which has additional constraints incorporated into the Bayes linear optimisation problem. For non-linear relationships between the latent variable and observations, Bayes linear will give a poor approximation. We propose a novel sequential Monte Carlo method for sequential inference on the network, which better copes with non-linear relationships. We give a method for simulating the random variables based upon the Bayes linear methodology. Finally, we look at the effect the ordering of the random variables has on the joint probability distribution of binary random variables, when they are simulated using this proposed Bayes linear method.

Acknowledgements

I would like to thank my supervisors, Paul Fearnhead and Kevin Glazebrook, for their encouragement and advice throughout my PhD. In particular, I would like to thank Paul for his invaluable knowledge and guidance. I would like to thank Ned Dimitrov for his time, insight and enthusiasm as well as for being an exceptionally welcoming host during my visits to the University of Texas at Austin and to the Naval Postgraduate School, California.

My PhD was funded by EPSRC as part of the STOR-i Doctoral Training Centre. I would like to thank everyone who is a part of STOR-i for helping to provide a supporting and stimulating learning environment. The experience would have been far less enjoyable without the friendships I have formed over the last four years. Finally, I would like to thank my friends and family for their continued support.

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree.

The actual word count for this thesis is 31105.

Lisa Jane Turner

Contents

List of Abbreviations

BL	Bayes Linear
BN	Bayesian Network
IS	Importance Sampling
KL	Kullback-Leibler
MCMC	Markov Chain Monte Carlo
MDP	Markov Decision Process
MRF	Markov Random Field
SMC	Sequential Monte Carlo
SIS	Sequential Importance Sampling
SIR	Sequential Importance Resampling
UCB	Upper Confidence Bound
VB	Variational Bayes

Chapter 1

Introduction

There are many applications where one may wish to search through a large set of items, such as emails or documents, to find a small subset of them which is relevant to a query. Often these items are distributed on edges of a network. We call the problem of finding relevant items distributed across the edges of a network a *network-based search*. Network-based search is increasingly common in many applications and contexts. We give motivating examples, which we return to later in the thesis.

The first example comes from corporate lawsuits. Companies are often required to hand over large databases of emails to one another and the information retrieved from these emails may make up part of their legal battle (Cavaliere et al., 2005). Many corporate cases rely on evidence from electronic communications such as the case against American Home Products who were charged with reckless indifference to human life after more than 33 million emails were searched to find emails relating to the charges (Volonino, 2003) and in Oracle's defence in *Oracle America, Inc. v. Google Inc.* where emails suggested Google was more aware that their use of Java APIs could infringe upon licensing agreements than was previously claimed (Mullin, 2016). To find and judge which of the emails are relevant to a case requires searching through a huge network of emails, the majority of which will be irrelevant, potentially in time pressured situations at the cost of millions of dollars and thousands of hours (Flynn and Kahn, 2003). Despite this, the courts often see the searching

of emails as not unduly burdonsome.

Also within the legal system, McGinnis and Pearce (2014) state legal search as one of the five main areas on the cusp of a machine intelligence invasion. Cases that have similar issues or facts as the current case are used to set a precedent for a trial. To find precedents, legal teams search through a huge number of cases looking for those with similar topics. Network analysis has been suggested as a method to find influential cases (Fowler et al., 2007), where the nodes are the cases and links relate to citations. An alternative would be to distribute cases on a network, with the people involved as the nodes and the cases on the edges – in a similar way as emails within a company. As well as searching through cases for similar topics using network-based search algorithms, the people who tend to work on the relevant cases could also be found. People who work on one relevant case may be more likely to work on another. Precedent search is similar to academic searches, and network-based search algorithms could be used as an alternative to those currently used which are based on words, citation counts etc. The use of a network-based search may better identify relevant authors and topics.

Another area to which network-based searches are applicable is intelligence processing. Efficient military operations and law enforcement rely on the timely processing of intelligence (Hughbank and Githens, 2010). An overwhelming amount of intelligence is collected daily, particularly communications intelligence, where the use of social media, text messaging and emails has drastically increased (Duyvesteyn et al., 2014). Many terrorist attacks could have been stopped or at least mitigated if the available intelligence was better processed and analysed (Gorman, 2008). One example is the Christmas Day bombing of Northwest Airlines Flight 253, for which 14 intelligence failures were reported (Select Committee on Intelligence, United State Senate, 2010) including failing to uncover key intelligence reports on the bomber. The National Intelligence Strategy (2009) states a key area of improvement is to narrow the gap between collecting intelligence and being able to make sense of the intelligence which is collected. A result of the overwhelming amount of intelligence collected is that often the processor is faced with a bottleneck of intelligence items; far more are collected than can be processed. In a time critical situation, the main challenge for the

processor is to provide the analysts with the largest set of relevant intelligence items in the given time window.

In these examples, we have a set of emails or other communications, henceforth called *items* for simplicity, and we wish to search this set of items to find those which are relevant to the query of interest. Each of the items involves two or more participants. The participants and the items between them induce a network where the participants form the nodes. An edge exists between two nodes if there is at least one item between the two associated participants. Participants can also be relevant or irrelevant to the query. Relevant items are more likely to occur between relevant participants. In addition, it is likely that relevant participants will cluster in groups. Thus, the network contains information that can be exploited to help decide which items to observe. Furthermore, each of the problems is time pressured and could involve huge amounts of possible items to search through. Hence, scalable search methods are required in order to be able to process as many items as possible in a short period of time and select which items to observe to maximise the chance of of finding relevant ones. The problem can be seen as having three related steps:

1. Constructing an appropriate joint prior distribution for the relevance of participants and items.
2. Updating this joint distribution as items are observed.
3. Deciding which item to observe next, given the current joint distribution on the relevance of items and participants.

The focus of this thesis is on developing inference methods for the second step, with methodology influenced by the other steps.

In Chapter 2 we discuss some of the literature on graphical models which can be used to model the random variables in the communication network. The structure of the graphical model helps minimise the computational cost of inference. Exact inference methods pass messages around the graph to break the problem down into smaller problems. The junction tree algorithm is the exact inference method is used in the thesis for the smaller networks.

For large networks, these exact inference methods become computationally challenging. The chapter also gives an overview of popular approximate inference methods commonly used in graphical models. In many cases, constructing a full prior specification can be difficult, we also introduce an alternative inference method (Bayes Linear) which only requires partial belief specification.

The problem of deciding which observation to make next, based on the current information, can be modelled as a sequential decision problem. In Chapter 3 we give a brief introduction to sequential decisions problems and discuss some of the large body of literature on the approaches used to solve these problems.

In Chapter 4 we introduce a model based on Bayes Linear methodology. Recent work by Dimitrov et al. (2016) has shown that using the information about the network structure together with a modelling assumption that relevant items and participants are likely to cluster together, can greatly increase the rate of screening relevant items. However their approach is computationally expensive and thus limited in applicability to small networks. Bayes Linear methods provide a natural approach to modelling such data; they output posterior summaries that are most relevant to heuristic policies for choosing future items and they can easily be applied to large-scale networks. For binary data, we introduce a new optimisation problem with additional constraints on the posterior summaries incorporated into the Bayes Linear optimisation problem. For a Bernoulli random variable, the mean will be in the range $[0, 1]$. The additional constraints we add ensure the updated expectations are also in this range. We show the resulting optimisation problem has an analytical solution. For a small network, we compare the performance of the constrained Bayes linear model with performance when exact inference is used, which is assumed to be the true model. We show that for networks formed from an email database, where there is no true underlying model, the Bayes linear model gives good results, and can be applied to networks which are too large for the exact inference models. The work in Chapter 4 makes up a paper which has been submitted for publication with co-authors Professor Paul Fearnhead and Dr Nedialko Dimitrov.

The Bayes Linear model developed in Chapter 4 uses a linear approximation to the relationship between the observations and the latent variables. The linear approximation will give a poor representation of highly non-linear relationships, so the Bayes Linear updates will give a poor approximation. Chapter 5 introduces an new, alternative approach to inference for sequential search on binary networks to overcome this problem. The method combines sequential Monte Carlo and Bayes Linear updates to approximate the joint distribution of the latent variables given observations; only random variables directly involved in observations are included in the Monte Carlo approximation. Building on the success of the Bayes Linear method in Chapter 4, Bayes Linear updates are used to both approximate the remaining latent variables and to approximate the probability of the realisations in the Monte Carlo approximation. Although this method is more computationally expensive than the Bayes Linear model, it does not require a linear approximation to the relationship between the latent variables and the observations. We show the method better approximates exact inference in the binary Markov random field, than simpler approximate inference methods. A comparison of the computational time for the three methods is given in Section 5.5.3.

The Bayes Linear methods from Chapter 4 and Chapter 5 can be used to define a joint distribution for binary random variables with a given mean and covariance. This distribution is specified by ordering the random variables, $Z_1, \ldots, Z_n$, and using Bayes Linear updates to calculate the distribution of Z_i given $Z_1, \ldots, Z_{i-1}$. We show in Chapter 6 that the ordering of the random variables affects the resulting joint distribution. Furthermore we investigate, both theoretically and empirically, which aspects of the ordering do/do not affect the joint distribution.

Finally, Chapter 7 we conclude with a discussion of the main contributions. A discussion of further possible paths of research is given.

Chapter 2

Inference in Graphical Models

2.1 Introduction to Graphical Models

Performing inference on a large number of random variables can be a computationally challenging problem. For example, to fully define a set of n binary random variables requires the specification of 2^n probabilities – one for each possible combination of the random variables. However, data associated with real-world phenomena often have structure, which can be exploited by representing the data through graphical models. The models provide an intuitive way to represent the random variables and visualise the relationship between these variables (Whittaker, 2009).

A *graph* $G = (V, E)$ is a set of *nodes*, $V = \{1, \ldots, n\}$, (also known as vertices) and a set of *edges*, E, (also known as links or arcs). An edge in the graph connects two nodes. A *graphical model* is a graph where the nodes represent random variables and the edges express probabilistic relationships between the random variables. Two random variables are said to be *conditionally independent* if the value of one variable does not directly impact the value of the other variable. For example, A and B are conditionally independent given C if $P(A \mid B, C) = P(A \mid C)$. Graphical models allow us to define conditional independence relationships between random variables and provide an effective approach to dealing

with the complexities of the data, as computations can be expressed in terms of graphical manipulations (Koller and Friedman, 2009).

The two major types of graphical models are Bayesian networks (also known as directed graphical models) and Markov Random Fields (also known as undirected graphical models). Section 2.1.1 gives a brief introduction to Bayesian networks, and describes the link between these and Markov Random Fields (MRFs). We mainly focus on the problem of inference in MRFs with discrete probability distributions associated with the random variables (discrete MRFs). Section 2.1.2 introduces MRFs and Section 2.2 describes how the structure of the graph can be exploited to perform inference in a MRF.

2.1.1 Bayesian Networks

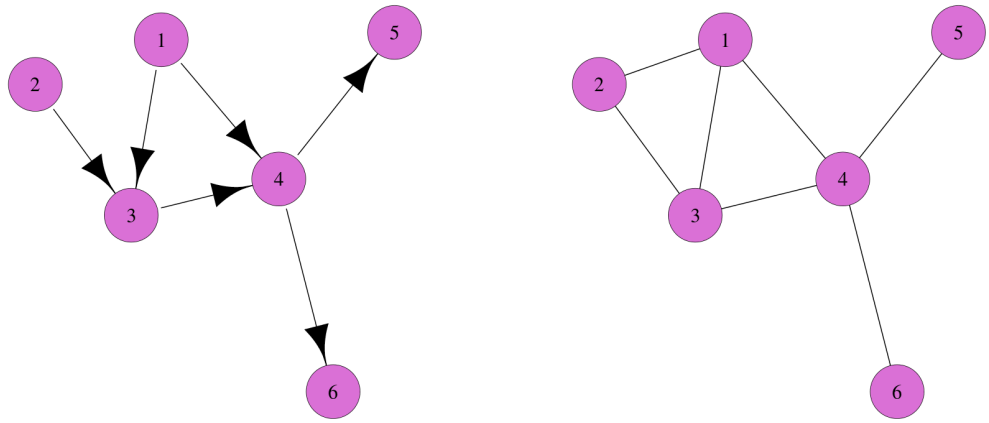
Bayesian networks are a type of graphical model where all edges in the graphical model are directional. They are useful when modelling phenomena where the conditional independence properties naturally have a direction. For example, Bayesian networks can be used in medical diagnostics to help infer the likelihood of different causes of illness given the symptoms (Miller et al., 1982; Nikovski, 2000), to model reactions in biological networks (Needham et al., 2007) and in the use of Turbo decoding problems in wireless 3G and 4G mobile telephony standards (Murphy et al., 1999; Berrou and Glavieux, 1996). The graph represents a factorisation of the joint probability over all random variables. For a set of random variables $\mathbf{X} = (X_1, \ldots, X_n)$, a Bayesian network is represented by the factorisation:

$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i | \mathbf{X}_{pa(i)}), \tag{2.1.1}$$

where $pa(i)$ are the parents of node i. The joint distribution of the Bayesian network in Figure 2.1.1a can be factorised using the conditional independences as:

$$p(X_1, \ldots, X_6) = p(X_6 \mid X_4)p(X_5 \mid X_4)p(X_4 \mid X_3, X_1)p(X_3 \mid X_2, X_1)p(X_2)p(X_1). \tag{2.1.2}$$

Classic models like hidden Markov models (Ghahramani, 2001) and neural networks are examples of Bayesian networks. Many of these have model specific inference methods, such as the forward-backward algorithm and Viterbi algorithm (Viterbi, 1967) for hidden Markov models. Generally, the exact inference methods discussed in Section 2.2 for MRFs are also inference methods for Bayesian networks, where the graph is first moralised to create an undirected network (the moral graph). The moral graph is formed by adding edges between all nodes which have a common child and making all the edges undirected. The MRF in Figure 2.1.1b is the moral graph for the Bayesian network in Figure 2.1.1a.



(a) Bayesian Network (b) Markov Random Field

Figure 2.1.1: Types of graphical models. The MRF in Figure 2.1.1b is the moral graph of the Bayesian network in Figure 2.1.1a.

2.1.2 Markov Random Fields

MRFs are useful when modelling a variety of phenomena where there is no natural directionality to the relationship between random variables. They are well-suited to make sense of large, complicated data sets which are increasingly common in the era of big data. This natural representation of data has lead to MRFs being widely used to represent data in many areas of statistics, physics, biology and machine learning. For example, MRFs are used in image modelling, where pixels are connected to neighbouring pixels (Geman and

Geman, 1984), to model spacial dependence between neighbouring regions in a map (Besag, 1974; Ripley, 2005) and in many problems in combinatorial optimisation which are defined in graph-theoretic terms, and thus are naturally recast in the graphical model formalism (Nemhauser and Wolsey, 1988; Maneva et al., 2007). The Ising model is an example of an MRF that arose from statistical physics. It was originally used for the modelling of magnetic dipole movements of atomic spins in a lattice graph, where each spin interacts with its neighbours. The model allows the identification of phase transitions as a simplified model of reality (Baxter, 1982). Many of the ideas behind the Bethe approximation to the Ising model (Bethe, 1935) are still applicable and have been shown to have links to widely used approximate variational inference methods, discussed in Section 2.3.1.

Let $G = (V, E)$ denote the graph associated with the set of random variables, $\mathbf{X} = (X_1, \ldots, X_n)$, where the nodes in the network represent the random variables and the edges represent direct dependencies. A *separating subset* for two random variables in the graph, X_u and X_v, is one where all paths between X_u and X_v pass through the subset. The set of random variables is said to form a *Markov Random Field* if X_u and X_v are independent, given a separating subset of nodes in the network (Bishop et al., 2006).

MRFs induce local dependencies rather than global dependencies, leading to computational advantages. For example, if the network in Figure 2.1.1b represents a MRF, then using the notation $\mathbf{X}_{-i} = (X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$, the full conditional of $P(X_1|\mathbf{X}_{-1})$ is equivalent to:

$$P(X_1|\mathbf{X}_{-1}) = P(X_1|X_2, X_3, X_4).$$

A *clique* in the graph is a subset of nodes such that there exists an edge between all pairs of nodes in that subset and a *maximal clique* is one which is not contained within any other clique. For two random variables a and b connected by an edge, the *factor* $\phi(a, b)$ describes the affinity between random variables; a probability distribution is also a factor but factors are not restricted by normalising constants. Using the Hammersley-Clifford theorem (Hammersley and Clifford, 1971) the joint distribution of the random variables

can be defined in terms of factors over the maximal cliques of the graph. This can lead to computational advantages as described below. Hence, the joint distribution of the random variables is given by:

$$P(\mathbf{X}) \propto \prod_{c \in \mathcal{C}} \phi_c(\mathbf{X}_c),$$

where $\mathcal{C}$ is the set of maximum cliques in G, and $\phi_c(\mathbf{X}_c)$ is the factor associated with maximal clique c. This representation of the random variables reduces the computational cost of inference and the memory required to specify the full distribution. For the binary MRF shown in Figure 2.1.1b, the naive specification of the data requires $2^6 = 64$ numbers. However, using the Hammersley-Clifford theorem the joint distribution can be specified as:

$$P(\mathbf{X}) \propto \phi_1(X_1, X_2, X_3)\phi_2(X_1, X_3, X_4)\phi_3(X_4, X_5)\phi_4(X_4, X_6),$$

requiring $2^3 + 2^3 + 2^2 + 2^2 = 24$ probabilities to specify the full distribution.

2.2 Exact Inference in Markov Random Fields

Graphical models can, in principle, be used to answer many types of queries. Often this involves calculating marginal or conditional probabilities of random variables in the model. The brute force method for inference in discrete MRFs requires the full joint specification of the random variables and ignores any conditional independence properties. As a result, the cost of inference grows exponentially with the number of nodes in the network. Instead, more efficient inference algorithm exist which involve passing messages around the graph, exploiting the conditional independence relationships in the graphical model and minimising the computational cost. These types of algorithms are known as message passing algorithms and are a type of dynamic programming. Dynamic programming is a method that is used for solving complex problems by breaking them down into sub-problems, and seeking to solve each sub-problem only once, reducing the number of computations (Smith, 1991).

Examples of message passing algorithms in special graphical models include the forward-backward algorithm and Viterbi algorithm for hidden Markov models and belief propagation for undirected trees, discussed in Section 2.2.1. Belief propagation forms the basis for the junction tree algorithm, described in Section 2.2.2. This is the exact inference method used within this thesis. Overviews of inference in graphical models and examples can be found in Bishop et al. (2006) and Koller and Friedman (2009).

2.2.1 Belief Propagation

A *tree* is a graph where there is only one path between any two pairs of nodes in the graph; there are no loops. Belief propagation (Pearl, 1982) is a message passing algorithm for inference on tree networks. The method forms the basis of the junction tree algorithm for exact inference in general graphical models and many of the approximate inference algorithms which work well for intractable or cyclic graphs, discussed in Section 2.3.1.

The method is an iterative procedure, where neighbouring variables in the tree pass messages to each other. For the undirected tree, $T = (V, E)$, associated with the random variables $X = (X_1, \ldots, X_n)$, a parameterisation for the joint distribution is:

$$p(\mathbf{X}) = \frac{1}{Z} \prod_{(i,j)\in E} \phi(X_i, X_j). \tag{2.2.1}$$

If $T_{i\to j}$ is as the sub-tree containing node i if node j is removed, and $X_{T_{i\to j}}$ are all nodes in this sub-tree, the message $M_{ij}(X_j)$, is defined as:

$$M_{ij}(X_j) = \sum_{X_{T_{i\to j}}} \phi(X_i, X_j) \prod_{(i',j')\in T_{i\to j}} \phi(X_{i'}, X_{j'}). \tag{2.2.2}$$

and can be thought of as the message that is passed from node i to node j. The messages can be recursively computed as:

$$M_{ij}(X_j) = \sum_{X_j} \phi(X_i, X_j) \prod_{k\in ne(i)\setminus j} M_{ki}(X_i), \tag{2.2.3}$$

where $ne(i)$ are the neighbouring nodes to node i. Thus, the marginal distribution of the random variable X_j is given by:

$$p(X_j) \propto \prod_{i \in ne(j)} M_{ij}(X_j). \tag{2.2.4}$$

The joint distribution over two random variables is:

$$p(X_i, X_j) \propto \phi(X_i, X_j) \prod_{k \in ne(i) \setminus j} M_{ki}(X_i) \prod_{l \in ne(j) \setminus i} M_{lj}(X_j). \tag{2.2.5}$$

Algorithm 1 gives the belief propagation algorithm for computing marginal distributions in undirected trees. For tree networks, the computational cost of belief propagation grows linearly with the number of nodes in the network.

So far, the case where all variables are hidden has been considered. In many cases, the marginal distribution when a subset of the variables are observed will be required. Suppose $\mathbf{X}$ is split into two subsets, $\mathbf{Z}$ and $\mathbf{Y}$, with observed values $\mathbf{Y} = \mathbf{y}$. Prior to computing the messages,the joint distribution $p(\mathbf{X})$ is multiplied by the product of identity functions $\prod_i \mathbb{1}(Y_i, y_i)$, where $\mathbb{1}(Y_i, y_i) = 1$ if $Y_i = y_i$ and zero otherwise. The product corresponds to $p(\mathbf{Z}, \mathbf{Y} = \mathbf{y})$ and hence is proportional to $p(\mathbf{Z} \mid \mathbf{Y} = \mathbf{y})$ which is the marginal distribution over hidden variables. The belief propagation algorithm can then be used to calculate the marginals where any summations over variables involving $\mathbf{Y}$ collapse to a single term.

Algorithm 1 Belief Propagation for Undirected Trees

1. Choose a root node
2. Pass messages from the leaves up to the root and then back down again, using

$$M_{ij}(X_j) = \sum_{X_j} \phi(X_i, X_j) \prod_{k \in ne(i) \setminus j} M_{ki}(X_i) \tag{2.2.6}$$

3. Given messages, compute required marginals.

2.2.2 Junction Tree Algorithm

There are many variants of the message passing algorithms for exact inference in graphical models. They all work by systematically exploiting the conditional independence properties encoded in the pattern of edges to compute marginal probabilities (Jordan, 2004). The structure of the graphical model means that some sub-expressions only depend on a small number of variables. Computing these once and caching them means the sub-expressions do not need to be calculated multiple times. We describe the junction tree algorithm by Lauritzen and Spiegelhalter (1988) for inference in MRFs. The method can also be applied to Bayesian networks by first moralising the graph. Other methods, such as conditioning (Shachter et al., 1994) and clique trees (Shafer and Shenoy, 1990), are computationally equivalent with some trade off. The sum product algorithm for factor graphs (Kschischang et al., 2001) uses two types of message in accordance with the two types of node in the factor graph and can be applied to slightly more general problems than the junction tree algorithm.

If loops are present in the graph then message passing, like that described in Section 2.2.1, can lead to overconfidence due to double counting of information and to non-convergent behaviour (Koller and Friedman, 2009); the marginal distributions will only be approximate. The junction tree algorithm prevents this by ensuring that different channels of information communicate with each other to prevent double counting. The general idea is to create a tree of cliques and perform message passing between these cliques.

Constructing the Junction Tree

The first step in the junction tree algorithm is to construct the junction tree from the graph. In order to do this, the graph must first be triangulated, and then the junction tree is constructed from the triangulated network. *Triangulation* adds edges to the network so that any chord-less cycles of a size greater than four are eliminated. For example, the network in Figure 2.2.1 can be triangulated by adding an edge between nodes 1 and 3 or

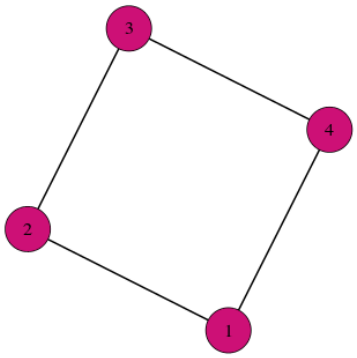


Figure 2.2.1: A chord-less cycle. The graph can be triangulated by adding the edge $(1, 3)$ or $(2, 4)$.

nodes 2 and 4. This removes conditional independence relationships, and enlarges the size of the distribution.

A junction tree is a tree where the nodes and edges are labelled with a set of variables. The variables on the nodes are the cliques in the triangulated network, and the variable sets on edges are called separators. The separators contain all random variables common to both sets of cliques associated with the involved nodes. A junction tree has two properties:

1. Cliques contain all adjacent separators.
2. If two cliques contain variable X, all cliques and separators on the path between the two cliques contain X (the running intersection property).

The running intersection property is required for consistency, so local consistency of marginal distributions between cliques and separators guarantees global consistency. This property is satisfied by considering cliques in the triangulated network rather than the original MRF. Converting a triangulated graph to a junction tree is done using Algorithm 2 (Bishop et al., 2006). This algorithm finds the maximum weight spanning tree of the weighted graph, where the nodes are the maximal cliques in the triangulated graph and edges exist between

nodes if they have common variables in the cliques. The weight on an edge represent the size of the separator associated with the adjacent cliques. The maximum weight spanning tree can be calculated using Kruskal's algorithm (Kruskal, 1956) by negating the weights on each edge.

Algorithm 2 Convert a Triangulated Graph to a Junction Tree

1. Find the set of maximal cliques $C_1, \ldots, C_k$.
2. Construct a weighted graph, with nodes labelled by the maximal cliques, and edges labelled by intersection of adjacent cliques.
3. Define the weight of an edge to be the size of the separator.
4. Run maximum weight spanning tree on the weighted graph to give the junction tree.

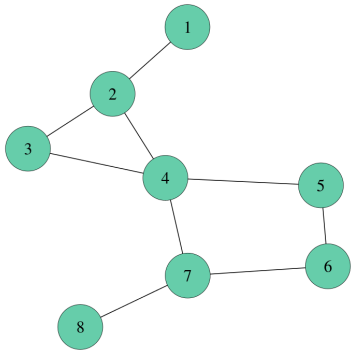
The maximal cliques in the triangulated graph correspond to nodes of the junction tree. Therefore we can write the joint distribution as a product over the nodes of the junction tree:

$$p(\mathbf{X}) = \frac{1}{Z} \prod_i \phi_{C_i}(\mathbf{X}_{C_i}), \tag{2.2.7}$$

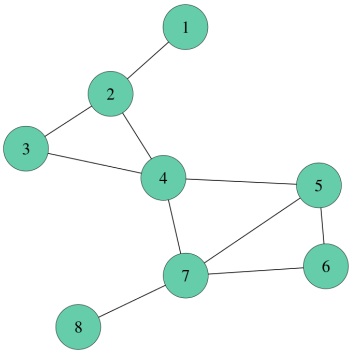
where C_i is the clique associated with node i of the junction tree. An example of constructing a junction tree from an MRF is given in Figure 2.2.2.

Message Passing in the Junction Tree

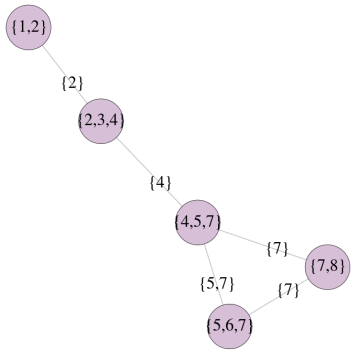
Messages are passed through the junction tree in a similar manner to belief propagation in a tree network. A clique in the junction tree is only allowed to send a message to a neighbouring clique j when it has received messages from all other neighbours. This is known as the *message passing protocol*. Define $S_{ij} = C_i \cap C_j$ as the separator between cliques i and j and let $T_{i \to j}$ be the union of cliques on the i side of j. The message passed



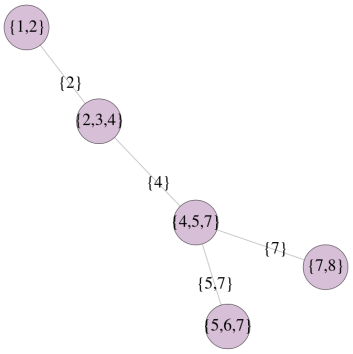
(a) MRF



(b) Triangulated Graph



(c) Maximal Clique Graph



(d) Junction Tree

Figure 2.2.2: Example of how a junction tree is calculated from an undirected network. Figure 2.2.2d shoes a possible junction tree for the MRF in Figure 2.2.2a.

from clique i to clique j in the junction tree algorithm are defined as:

$$M_{ij}(X_{S_{ij}}) = \sum_{X_{T_{i\to j}\setminus C_j}} \prod_{k:C_k\subset T_{i\to j}} \phi_k(X_{C_k}). \tag{2.2.8}$$

Defining a root node in the junction tree, and then starting at the leaf nodes, the messages can be computed recursively. The message $M_{ij}(X_{S_{ij}})$ in (2.2.8) is equivalent to:

$$M_{ij}(X_{S_{ij}}) = \sum_{X_{C_i}\setminus S_{ij}} \phi(X_{C_i}) \prod_{k\in ne(i)\setminus j} M_{ki}(X_{S_{ki}}). \tag{2.2.9}$$

On completion of a sweep of message passing from the leaf of the tree up to the root and then back down again, all marginal probabilities can be calculated from the messages where the marginal distributions for the cliques i is:

$$p(X_{C_i}) \propto \phi(X_{C_i}) \prod_{k \in ne(i)} M_{ki}(X_{S_{ki}}). \tag{2.2.10}$$

The junction tree algorithm for inference in MRFs is given in Algorithm 3. Observations can be incorporated into the algorithm by multiplying a single clique containing the involved random variables by the likelihood of the observation and completing a sweep on the message passing (Koller and Friedman, 2009).

Algorithm 3 Junction Tree Algorithm

Setup:

1. Find the triangulated network
2. Find the junction tree using Algorithm 2

Run:

1. Choose a root node.
2. Pass messages from the leaves up to the root and then back down again, using (2.2.9).
3. Calculate desired marginals.

Computational Cost

The junction tree algorithm works by systematically exploiting the structure of the graphical model. The majority of the computational cost comes from the message passing stage. However, the time and space complexity is dominated by the size of the largest clique in the junction tree; the computational cost grows exponentially with the tree width of the network. The tree width of the network is the size of the largest clique in the triangulated network minus one. Finding a triangulation which minimises the size of the largest clique can have a large effect on the efficiency of exact inference. The problem of finding the optimal tree width of a graph is NP-hard (Arnborg et al., 1987) so a lot of research has gone into efficient approximate algorithms (Amir, 2001; Becker and Geiger, 1996; Robertson and

Seymour, 1986).

In the best case, the tree width is small and the triangulated network which achieves this tree width is easy to find. In these cases, the junction tree algorithm provides a fast exact method for inference in graphical models (see Wainwright and Jordan (2008) for examples). However, in many cases, the graphical model is too complex and the tree width is too large for exact inference to be viable. In these cases, we rely upon approximate methods to perform inference in the graphical model.

2.3 Approximate Inference Methods for Graphical Models

A key obstacle in Bayesian inference is calculating marginal and posterior distributions. In an ideal setting, computing these can be done analytically, such as using the the junction tree algorithm for random variables in a graphical model. For large, complex data the computational costs associated with exact inference is too large. Instead, approximate inference algorithms are used to estimate the quantities of interest.

Exact inference algorithms can act as a starting point for approximate methods. These types of methods hope to use the structure of the graphical model to aid performance. They exploit situations where nodes or clusters of nodes are nearly conditionally independent, such as when node probabilities are well determined by only a subset of neighbouring nodes (Jordan et al., 1999). Pruning algorithms (Kjærulff, 1994), search based methods (Henrion, 1991) and localised partial evaluation (Draper and Hanks, 1994) have all been suggested to deal with these situations. Their advantage is that they still use the conditional independence properties exploited in the exact inference methods. However, they still suffer from many of the problems associated with exact inference, like the computational cost, which make exact inference difficult (Jordan et al., 1999).

Section 2.3.1 and 2.3.2 give an overview of alternative approximate inference methods which are popular methods for inference in graphical models. Variational methods (Section 2.3.1) have emerged as fast inference methods for graphical models and an alternative to slower

Markov Chain Monte Carlo (MCMC) methods (Section 2.3.2). Both variational methods and Monte Carlo methods require the full specification of prior beliefs. In Section 2.3.3 we introduce Bayes linear methodology which can be used as an approximate inference technique when only a partial prior specification is required.

2.3.1 Variational Methods

Variational methods (Jordan et al., 1999; Wainwright and Jordan, 2008) are a popular group of approximate inference methods for graphical models. The methods can be applied to general probabilistic inference problems but their main application is in graphical models. The basic idea of variational methods is to characterise a probability distribution as the solution to a high dimensional optimisation problem, alter this optimisation problem and solve the altered optimisation problem.

Variational Bayes (VB) is a variational method for approximating the true distribution, $p(\mathbf{X})$, with the approximating distribution, $q(\mathbf{X})$, where the closeness of the two distributions is measured through the Kullback-Leibler (KL) divergence. For a set of latent variables $\mathbf{X}$ and observations $\mathbf{Y}$, the interest may lie in the posterior distribution of the unknown variables given the observations:

$$p(\mathbf{X} \mid \mathbf{Y}) = \frac{p(\mathbf{X})p(\mathbf{Y} \mid \mathbf{X})}{p(\mathbf{Y})} \tag{2.3.1}$$

The VB approximation to the conditional distribution, $p(\mathbf{X} \mid \mathbf{Y})$ assumes a family of distributions, $q(\mathbf{X}) \in \mathcal{D}$ and approximates the distribution $p(\mathbf{X} \mid \mathbf{Y})$ by searching for $q^*(\mathbf{X}) \in \mathcal{D}$ which minimises the KL divergence:

$$KL(q(\mathbf{X}) \,||\, p(\mathbf{X} \mid \mathbf{Y})) = \mathbb{E}_q[\log(q(\mathbf{X})] - \mathbb{E}_q[\log(p(\mathbf{X} \mid \mathbf{Y}))]. \tag{2.3.2}$$

Wainwright and Jordan (2008) consider restricting the family of distributions to the exponential family and searching for the optimal distribution within this family. VB can be seen as an inner approximation which approximates the posterior with a distribution that can be

used for inference. The complexity of the family of distributions determines the complexity of the optimisation; it is more difficult to approximate over a complex family than a simple family. As a result, a common variational family is the mean fields variational family, were the latent variables are assumed to be mutually independent:

$$q(\mathbf{X}) = \prod_i q_i(X_i). \tag{2.3.3}$$

The idea for the mean fields approximation was adapted from statistical physics for inference in graphical model (Jordan et al., 1999; Attias, 1999).

The KL divergence, $KL(q \mid p)$, is mainly considered for the computational advantages and can be seen to prefer q in areas of high probability under q. Hence, drawing samples and integrating with respect to q should be accurate but VB approximation may miss out on areas of high probability under p. Furthermore, whilst q will give a good approximation to the joint distribution, the marginals $q_i(X_i)$ should not be expected to resemble the true marginals $p_i(x_i)$ (Bishop et al., 2006). The approximation can be extended to include dependencies between random variables, resulting in structured variational inference (Saul and Jordan, 1996).

The optimisation problems are typically solved using co-ordinate gradient ascent type algorithms. These methods iterate between evaluating each data point and estimating the hidden structure. One of the most commonly used algorithms for solving the variational optimisation problem is the co-ordinate ascent mean-field variational inference algorithm (Blei et al., 2016; Winn and Bishop, 2005), which iteratively optimises each factor of the mean field variational distribution whilst holding the others fixed, finding a local optimal solution. The algorithm can be seen as a message passing algorithm; updating a random variable's variational parameters based on the variational parameters of the neighbouring nodes in the MRF. Co-ordinate descent type algorithms requires a full pass through the data for each iteration and can be time consuming for large datasets. To overcome this issue, stochastic variational methods (Hoffman et al., 2013) only evaluate a small number of data points at each iteration.

An alternative class of algorithms is obtained by considering an outer approximation, where the set of parameters must satisfy a number of consistency relationships that arise from local neighbourhood relationships in the graph. A well used outer approximation is the belief propagation method in Section 2.2.1 when applied to graphs with cycles; called loopy belief propagation. It is a popular method due to its often remarkably accurate results (Murphy et al., 1999; McEliece et al., 1998). Loopy belief propagation is equivalent to a fixed point minimisation of the Bethe free energy equations for the graph (Yedidia et al., 2003). The Bethe approximation involves only those consistency relationships that rise from local neighbourhood relationships in the graphical model. There is no guarantee this algorithm will converge, and a large amount of theoretical and behavioural analysis has gone into understanding loopy belief propagation. Murphy et al. (1999) suggests that certain conditions, the approximation can often oscillate rather than converge. Koller and Friedman (2009) suggest several ways to help the loopy belief propagation work better. Algorithms known as cluster variational methods have been proposed to extend the Bethe approximation to higher order clusters of variables (Yedidia et al., 2003) along with other higher order variational methods (Leisink and Kappen, 2002; Minka, 2001).

Variational methods can be seen as more of an art than a method (Jordan et al., 1999) as there is a lack of understanding regarding which variational method should be applied in which case and no easy way to compare the effectiveness of the different methods. Often the choice has to be considered on case by case basis. Furthermore, theoretical analysis of variational inference is still relatively unexplored (Blei et al., 2016), although initial steps have been taken toward the analysis of convergence (Tatikonda and Jordan, 2002). Despite the lack of understanding of the convergence properties of loopy belief propagation, it has been successfully applied to various problems, such as stereo vision (Sun et al., 2003) and turbo decoding problems used in wireless 3G and 4G mobile telephony standards (McEliece et al., 1998).

2.3.2 Monte Carlo Methods

Monte Carlo methods rely on repeated random sampling from the distribution of interest to generate approximations to the true density. Consider a possibly multidimensional random variable, $\mathbf{X}$, with probability mass function or probability density function, $p(\mathbf{x})$, which is greater than zero for a set of values $\mathcal{X}$. The expected value of some measurable function $g : \mathbf{X} \to \mathbb{R}^d$, with respect to the probability density $p(\mathbf{x})$ is:

$$E[g(\mathbf{X})] = \int_{\mathcal{X}} g(\mathbf{x})p(\mathbf{x})d\mathbf{x}, \tag{2.3.4}$$

if $\mathbf{X}$ is continuous. The integral is replaced by a summation if the probability distribution is discrete. For a wide class of problems, it is not possible to evaluate the integral (or summation) analytically. Monte Carlo methods represent a class of techniques where samples, simulated from the target distribution, can be used to approximate (2.3.4). Perfect Monte Carlo assumes samples can be drawn directly from the target distribution. If this is possible, taking n independent and identically distributed samples of $\mathbf{X}$, $\left\{\mathbf{x}^{(i)}\right\}_{i=1}^{N}$, from $p(\mathbf{X})$, the Monte Carlo estimate of the expectation of $g(\mathbf{x})$ is:

$$\hat{g}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} g(\mathbf{x}^{(i)}). \tag{2.3.5}$$

Monte Carlo methods are a simple appealing approximation technique with good statistical properties (Robert, 2004). The Monte Carlo estimator, $\hat{g}(\mathbf{X})$ is an unbiased estimator of $E[g(\mathbf{X})]$. Using the law of large numbers, $\hat{g}(\mathbf{x})$ will converge almost surely to the $E[g(\mathbf{X})]$ as $N \to \infty$. Furthermore, the variance:

$$\mathrm{Var}(\hat{g}(\mathbf{X})) = \mathrm{Var}(\frac{1}{N}\sum_{i=1}^{N} g(\mathbf{X}^{(i)})) = \frac{\mathrm{Var}(g(\mathbf{X}))}{N}, \tag{2.3.6}$$

exists, then it decreases as the number of samples increases and does not depend on the dimension of $\mathbf{X}$. The literature on Monte Carlo methods is extensive, see Robert (2004) and Liu (2008) for further details. The remainder of this chapter gives an overview of Monte Carlo methods which are commonly used for inference in graphical models and, in

particular, those that are used in this thesis for inference.

Importance Sampling

In many situations, it is not possible to sample directly from the target distribution. This problem can be avoided by using an alternative sampling scheme. Importance sampling (IS) is a Monte Carlo method for evaluating integrals by drawing from a proposal distribution $q(\mathbf{x})$. The expectation of a function $g(\mathbf{x})$, over the target distribution can be calculated over the proposal instead. The proposal is chosen to be easy to sample from, and must satisfy that if $p(\mathbf{x}) > 0$ then $q(\mathbf{x}) > 0$. The samples are weighted based on their fit to the target distribution; those with larger weights are more influential in the Monte Carlo approximation.

Using a proposal distribution (2.3.4) can be written as:

$$\begin{aligned} E[g(\mathbf{X})] &= \int_{\mathcal{X}} g(\mathbf{x})p(\mathbf{x})d\mathbf{x}, \\ &= \int_{\mathcal{X}} g(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}q(\mathbf{x})d\mathbf{x}, \\ &= \int_{\mathcal{X}} g(\mathbf{x})w(\mathbf{x})q(\mathbf{x})d\mathbf{x}, \\ &= E[w(\mathbf{X})g(\mathbf{X})], \end{aligned} \tag{2.3.7}$$

which is the expectation of $w(\mathbf{X})g(\mathbf{X})$ with respect to the probability density $q(\mathbf{x})$ and $w(\mathbf{X}) = p(\mathbf{X})/q(\mathbf{X})$ is the importance weight. Sampling $\left\{\mathbf{x}^{(i)}\right\}_{i=1}^{N}$ from the proposal distribution, the Monte Carlo approximation to $E[g(\mathbf{x})]$ is:

$$\hat{g}(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} w(\mathbf{x}^{(i)})g(\mathbf{x}^{(i)}). \tag{2.3.8}$$

Often, the target distribution is only known up to a constant of proportionality, $p(\mathbf{x}) = \frac{\tilde{p}(\mathbf{x})}{Z}$,

where Z is the unknown normalising constant. In this case:

$$\begin{aligned} E[g(\mathbf{x})] &= \int g(\mathbf{x}) \frac{\tilde{p}(\mathbf{x})}{Zq(\mathbf{x})} q(\mathbf{x}) d\mathbf{x} \\ &\approx \frac{1}{ZN} \sum_{i=1}^{N} \tilde{w}^{(i)} g(\mathbf{x}^{(i)}) \end{aligned} \tag{2.3.9}$$

where $w^{(i)} = \frac{\tilde{p}(\mathbf{x}^{(i)})}{q(\mathbf{x}^{(i)})}$ and the normalising constant is approximated using the Monte Carlo sample as:

$$Z \approx \sum_{i=1}^{N} \tilde{w}^{(i)}. \tag{2.3.10}$$

Using this, the normalised importance samples are:

$$\hat{g}(\mathbf{x}) = \frac{1}{N} \sum_{i=1}^{N} w^{(i)} g(\mathbf{x}^{(i)}) \tag{2.3.11}$$

where:

$$w^{(i)} = \frac{\tilde{w}^{(i)}}{\sum_{j=1}^{N} \tilde{w}^{(j)}} \ , \ i = 1, \ldots, N. \tag{2.3.12}$$

By the law of large numbers, (2.3.11) converges to $E[g(\mathbf{x})]$. However, the Monte Carlo estimate is biased, with decreasing bias as the sample size increases. The choice of proposal is important, as the variance of the estimator is dependent on the choice of proposal distribution (see Robert (2004) for information on the optimal proposal).

Monte Carlo Markov Chain Methods

MCMC methods are an alternative approach to importance sampling. They produce samples from a Markov chain where the stationary distribution of the chain is the target distribution. This approach no longer creates independent and identically distributed samples. However, if the chain is sampled for a long time, hopefully enough independent samples can be collected from the target distribution and these used to approximate the distribution of

interest. The method has been well studied and applied to several areas of statistics (Liu, 2008). A justification of this approach, and corresponding convergence properties can be found in Robert and Casella (2013).

A Markov chain is a collection of dependent samples, $\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots\right\}$, where the probability of $\mathbf{x}^{(n+1)}$, given the samples at previous time steps, is only dependent on $\mathbf{x}^{(n)}$:

$$p(\mathbf{x}^{(n+1)} \mid \mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}) = p(\mathbf{x}^{(n+1)} \mid \mathbf{x}^{(n)}) = T(\mathbf{x}^{(n)}, \mathbf{x}^{(n+1)}) \tag{2.3.13}$$

and $T(\cdot, \cdot)$ is known as the transition kernel. The target distribution is said to be the invariant distribution if the transition kernel satisfies:

$$\sum_{x^*} T(\mathbf{x}^*, \mathbf{x}) p(\mathbf{x}^*) = p(\mathbf{x}). \tag{2.3.14}$$

In this case, if the chain converges it will converge to the target distribution. Simulation from the Markov chain requires an initial distribution $p(\mathbf{x}^{(0)})$. After a suitable burn in period, the samples from the Markov chain are dependent samples from $p(\mathbf{x})$. If the Markov chain is irreducible and aperiodic, then:

$$\frac{1}{N} \sum_{i=1}^{N} g(\mathbf{x}^{(i)}) \to E[g(\mathbf{X})] \tag{2.3.15}$$

with probability one as $N \to \infty$.

The Metropolis-Hastings algorithm (Hastings, 1970) is a method to generate a Markov chain that will converge to $p(\mathbf{x})$ by successively sampling from an arbitrary proposal distribution $q(\mathbf{x}^* \mid \mathbf{x})$ and imposing random rejection steps at each transition. Let $\mathbf{x}^{(n)}$ be the state of the MCMC algorithm after iteration n. To simulate the state at iteration $n+1$ a transition candidate $\mathbf{x}^* \sim q(\mathbf{x}^* \mid \mathbf{x}^{(n)})$ is simulated and with probability $\alpha(\mathbf{x}^{(n)}, \mathbf{x}^*)$:

$$\alpha(\mathbf{x}^{(n)}, \mathbf{x}^*) = \min \left\{ 1, \frac{q(\mathbf{x}^{(n)} \mid \mathbf{x}^*) \tilde{p}(\mathbf{x}^*)}{q(\mathbf{x}^* \mid \mathbf{x}^{(n)}) \tilde{p}(\mathbf{x}^{(n)})} \right\} \tag{2.3.16}$$

the transition candidate is accepted to give $\mathbf{x}^{(n+1)} = \mathbf{x}^*$ and $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)}$ otherwise.

The original Metropolis algorithm (Metropolis et al., 1953) was based on symmetry in the proposal distribution so that $q(\mathbf{x} \mid \mathbf{x}^*) = q(\mathbf{x}^* \mid \mathbf{x})$. The advantage of the Metropolis-Hastings algorithm is that the target distribution is only required up to a normalising constant since this cancels out in the acceptance probability anyway. If the proposal, $q(\mathbf{x} \mid \mathbf{x}^*)$, is chosen to be positive everywhere, then the convergence to $p(\mathbf{x})$ is guaranteed (Robert, 2004). However, the rate of convergence depends on the relationship between $q(\mathbf{x} \mid \mathbf{x}^*)$ and $p(\mathbf{x})$; choosing a $q(\mathbf{x} \mid \mathbf{x}^*)$ which minimises the autocovariance results in a more accurate estimator with a finite sample.

The Gibbs sampler (Geman and Geman, 1984) is a method of simulating a Markov chain, by successively sampling from the full conditional distributions of the components $p(x_i \mid \mathbf{x}_{-i})$ where $\mathbf{x}_{-i}$ denotes all components of $\mathbf{x}$ apart from x_i. The Gibbs sampler is a special case of the Metropolis-Hastings algorithm where the acceptance probability is 1. At each iteration, each component of $\mathbf{x}^{(n)}$ is updated. The ordering of the components can be random or deterministic (Liu, 2008). The difficulty in the method comes when the full conditionals are not of a standard form. Sampling methods, such as rejection sampling, can be used but often a Metropolis-Hastings sampler is used to provide the sample. Gibbs samplers are particularly applicable to problems of inference in graphical models. Using a MRF results in the full conditionals simplifying to conditionals dependent on only neighbouring nodes in the MRF and reduces the computation cost of computing them. A Gibbs sampler can be set up automatically from the graphical model specification (Gilks et al., 1994). Although the application of the Gibbs sampler naturally lends itself to inference in graphical model, the convergence of these types of models can be very poor. For some examples, where mixing is exponentially slow see Gore and Jerrum (1999).

Generally, MCMC algorithms cannot be applied for parameter inference in MRF as the normalising constant is doubly intractable. This problem has led to a body of MCMC methods (Murray et al., 2012; Caimo and Friel, 2011; Møller et al., 2006) to deal with doubly intractable likelihoods, most of which involve augmenting the posterior density so that the augmented posterior density can be easily sampled from. Alternative methods avoid the intractable likelihood problem by taking approximations to the normalising constant by

a pseudolikelihood which is the product of normalised distributions over lower dimensional problems (Friel et al., 2009).

MCMC methods are computationally intensive methods. When a large computational cost is not an issue, MCMC methods can provide accurate samples from the posterior distribution, providing asymptotically exact samples from the target distribution with convergence guarantees. However, the methods are less applicable to applications when the posterior distribution needs to be calculated quickly.

Sequential Monte Carlo Methods

The most widely used type of Monte Carlo methods are MCMC algorithms which can be applied to estimate the latent space $\mathbf{x}$ conditional on observations $\mathbf{y}$. However, consider the situation where the aim is to track a sequence of distributions and estimate the posterior distribution. This type of online inference problem is not particularly well suited to MCMC methods. Each new observation would require simulating an entire new Markov chain for the new posterior distribution. Instead the samples from the posterior for $\mathbf{x}_{1:t-1}$ within an algorithm can be used to attempt to approximately samples from the posterior for $\mathbf{x}_{1:t}$. Sequential Monte Carlo (SMC) methods can be used to do this by approximating a sequence of probability distributions on a sequence of probability spaces. The Monte Carlo techniques behind SMC have existed since the 1950s (Hammersley and Morton, 1954) but a lack of computational power at the time and problems with degeneracy meant these methods were generally overlooked. Since the introduction of the bootstrap filter (Gordon et al., 1993) and more general resampling schemes, there has been a large increase in research in this area. SMC works by taking an empirical approximation of the density using a collection of samples (also known as particles) and corresponding weights. The typical application is for inference in state space models, see Doucet and Johansen (2009) for applications of this type where the SMC models are more commonly known as particle filters.

The latent variables $\{\mathbf{x}_t : t \in \mathbb{N}\}$, $\mathbf{x}_t \in \mathcal{X}$, are modelled as a Markov process with initial

distribution $p(\mathbf{x}_0)$ and transition probabilities $p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$:

$$p(\mathbf{x}_{1:t}) = p(\mathbf{x}_0) \prod_{i=1}^{t} p(\mathbf{x}_i \mid \mathbf{x}_{i-1}). \tag{2.3.17}$$

The observations at time t, are assumed to be conditionally independent given $\mathbf{x}_t$ and the joint distribution over both latent variables and observations is:

$$p(\mathbf{x}_{1:t}, \mathbf{y}_{1:t}) = p(\mathbf{x}_0) \prod_{i=1}^{t} p(\mathbf{x}_i \mid \mathbf{x}_{i-1}) \prod_{i=1}^{t} p(\mathbf{y}_i \mid \mathbf{x}_i). \tag{2.3.18}$$

The aim is to recursively estimate the posterior distribution $p(\mathbf{x}_t \mid \mathbf{y}_{1:t})$ using a Monte Carlo approximation:

$$\hat{p}(\mathbf{x}_t \mid \mathbf{y}_{1:t}) = \sum_{i=1}^{N} w_t^{(i)} \delta_{\mathbf{x}_t^{(i)}}(\mathbf{x}_t), \tag{2.3.19}$$

where $\delta(\cdot)$ is the Dirac delta function and $\left\{ w_t^{(i)}, \mathbf{x}_t^{(i)} \right\}_{i=1}^{N}$ is the Monte Carlo approximation.

Sequential Importance Sampling (SIS) is one way to sequentially calculate the Monte Carlo approximation. The approximation to the density is recursively updated as the new observations are received, without modifying the previously simulated states. SIS propagates the particles in the Monte Carlo approximation from time t to $t+1$ and updates the weights accordingly, to give an approximation to the posterior density at time $t+1$. Suppose there is a proposal distribution $q(\mathbf{x}_{0:t}|\mathbf{y}_{0:t})$, which is easy to sample from, that has the property $p(\mathbf{x}_{0:t}|\mathbf{y}_{0:t}) > 0 \;\Rightarrow\; q(\mathbf{x}_{0:t}|\mathbf{y}_{0:t}) > 0$ and can be updated recursively in time so that:

$$q(\mathbf{x}_{0:t+1}|\mathbf{y}_{0:t+1}) = q(\mathbf{x}_{0:t}|\mathbf{y}_{0:t}) q(\mathbf{x}_{t+1}|\mathbf{x}_{0:t}, \mathbf{y}_{0:t+1}). \tag{2.3.20}$$

Using this proposal distribution, the Monte Carlo approximation at time t can be propagated to time $t+1$ without modifying the past. The joint distribution at time $t+1$ up to a constant

of proportionality is:

$$p(\mathbf{x}_{t+1}|\mathbf{y}_{0:t+1}) \propto \int p(\mathbf{y}_{t+1}|\mathbf{x}_{t+1})p(\mathbf{x}_t|\mathbf{y}_{0:t})p(\mathbf{x}_{t+1}|\mathbf{x}_t)d\mathbf{x}_t,$$
$$= \int \frac{p(\mathbf{x}_t|\mathbf{y}_{0:t})q(\mathbf{x}_{0:t}|\mathbf{y}_{0:t})}{q(\mathbf{x}_{0:t}|\mathbf{y}_{0:t})} \frac{p(\mathbf{y}_{t+1}|\mathbf{x}_{t+1})p(\mathbf{x}_{t+1}|\mathbf{x}_t)q(\mathbf{x}_{t+1}|\mathbf{x}_{0:t},\mathbf{y}_{0:t+1})}{p(\mathbf{y}_{t+1}|\mathbf{y}_{0:t})q(\mathbf{x}_{t+1}|\mathbf{x}_{0:t},\mathbf{y}_{0:t+1})} d\mathbf{x}_t \tag{2.3.21}$$

Assuming there is a weighted Monte Carlo approximation at time t, $\left\{w_t^{(i)}, \mathbf{x}_t^{(i)}\right\}_{i=1}^N$ to $p(\mathbf{x}_t|\mathbf{y}_{0:t})$ then (2.3.21) can be approximated by propagating the set of particles to give $\left\{\mathbf{x}_{t+1}^i\right\}_{i=1}^N$ and unnormalised weights:

$$\tilde{w}_{t+1}^{(i)} = \tilde{w}_t^{(i)} \frac{p(\mathbf{y}_{t+1}|\mathbf{x}_{t+1})p(\mathbf{x}_{t+1}|\mathbf{x}_t)}{q(\mathbf{x}_{t+1}|\mathbf{x}_{0:t},\mathbf{y}_{0:t+1})} \text{ , for } i = 1,\ldots,N. \tag{2.3.22}$$

The normalised weights are given by:

$$w_{t+1}^{(i)} = \frac{\tilde{w}_{t+1}^{(i)}}{\sum_{j=1}^N \tilde{w}_{t+1}^{(j)}},$$

providing a weighted particle approximation to the marginal distribution at time $t+1$. The joint distribution to $\{\mathbf{x}_{0:t+1}\}$ is approximated by storing the path of the particles $\mathbf{x}_{0:t+1} = (\mathbf{x}_{0:t}, x_{t+1})$ and still updating the weighting using (2.3.22).

Using the type of proposal distribution in (2.3.20), the weights of the particles can be sequentially updated in time, as the next observation becomes available. Assume it is possible to sample from the proposal distribution and both the likelihood and transitional probability are known. All that is needed is an initial set of samples to be generated. The proposal weights can be calculated sequentially. The SIS algorithm recursively propagates weights and particles as observation are received sequentially; it is given in Algorithm 4.

The choice of proposal distribution is important. The variance of the particle weights can only increase (Kong et al., 1994) for a proposal distribution of the form in (2.3.20). The variance of the particle weights increasing over time has a harmful effect on the accuracy and leads to the degeneracy phenomenon. Nearly all but a few particles will have negligible

Algorithm 4 The Sequential Importance Sampling Algorithm

1. At time $t = 0$:
   (a) For $i = 1, \ldots, N$
      i. Sample $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$
      ii. Evaluate the importance weights up to a normalising constant:

$$\tilde{w}_0^{(i)} = p(\mathbf{y}_0|\mathbf{x}_0^{(i)})$$

   (b) For $i = 1, \ldots, N$ normalise the importance weights:

$$w_0^{(i)} = \frac{\tilde{w}_0^{(i)}}{\sum_{j=1}^{N} \tilde{w}_0^{(j)}}$$

2. For t in 1 to T
   (a) For $i = 1, \ldots, N$
      i. Sample $\mathbf{x}_t^{(i)} \sim q(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_{0:t})$ and $\mathbf{x}_{0:t}^{(i)} = (\mathbf{x}_{0:t-1}^{(i)}, \mathbf{x}_t^{(i)})$
      ii. Evaluate the importance weights up to a normalising constant:

$$\tilde{w}_t^{(i)} = \tilde{w}_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) p(x_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{0:t})}$$

   (b) For $i = 1, \ldots, N$ normalise the importance weights:

$$w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}$$

weight after running some iterations; a lot of work is devoted to updating particles which have almost zero weight. The degeneracy phenomenon is a big problem in SMC. A good choice of proposal distribution is key to slowing down degeneracy and can be found by minimising the variance of the particle weights. The optimal proposal distribution, first given by Zaritskii et al. (1975) is:

$$\begin{aligned} q(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_{0:t})_{opt} &= p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t), \\ &= \frac{p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{x}_{t-1}^{(i)}) p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)})}. \end{aligned} \tag{2.3.23}$$

Substituting this into (2.3.20) gives:

$$w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)}). \tag{2.3.24}$$

This means that the importance weights at time t can be calculated, and the particles possibly resampled (with probability given by the weights), before the particles are propagated to time t. However, to use (2.3.23) it must be possible to sample from $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_t)$ and calculate $p(\mathbf{y}_t|\mathbf{x}_{t-1}^{(i)})$, one or both of which are often not possible (Doucet et al., 2000). A suboptimal choice of importance function must be used. The most popular choice is the transitional prior, $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, which gives a weight $w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t)$. It is no longer possible to calculate the importance weights before the particles have been propagated at time t. The auxiliary particle filter (Pitt and Shephard, 1999) tries to overcome this problem. At time $t-1$, the auxiliary particle filter tries to predict which samples will be in regions of high probability masses at time t by sampling each particle at time $t-1$ and propagating the sample forward to time t.

Degeneracy can also be minimised through a good choice of importance function and by including a resampling step in the SIS algorithm. The Sequential Importance Resampling (SIR), also known as the bootstrap filter (Gordon et al., 1993), was developed separately from the SIS algorithm and contains a resampling step at each iteration of the SIS algorithm. Since then, resampling has been shown to have both major practical and theoretical benefits (Doucet and Johansen, 2009). An IS approximation of the target distribution is based on weighted samples from $q(\mathbf{x}_{t-1}|\mathbf{y}_{0:t-1})$ and hence is not distributed according to $p(\mathbf{x}_t|\mathbf{y}_{0:t})$. Resampling provides a way to give an approximation from the target distribution by resampling N particles from the IS approximation. Each particle is picked with a probability proportional to the weight associated with it. Particles with low weight have a high probability of being removed and this results in multiple copies of particles with high weights. The future of these can then be explored independently. This provides stability in the future but at the cost of an increase in the immediate Monte Carlo variance (Doucet and Johansen, 2009). The SIR algorithm is given by Algorithm 5.

Algorithm 5 The Sequential Importance Resampling Algorithm

1. At time $t = 0$:
   (a) For $i = 1, \ldots, N$
      i. Sample $\mathbf{x}_0^{(i)} \sim p(\mathbf{x}_0)$
      ii. Assign weights $\tilde{w}_0^{(i)} = p(y_0|\mathbf{x}_0^{(i)})$
   (b) For $i = 1, \ldots, N$
      i. Normalise weights $w_0^{(i)} = \frac{\tilde{w}_0^{(i)}}{\sum_{j=1}^{N} \tilde{w}_0^{(j)}}$
2. For t in 1 to T
   (a) For $i = 1, \ldots, N$
      i. Resample $\tilde{\mathbf{x}}_{t-1}^{(i)}$ by resampling from $\left\{\mathbf{x}_{t-1}^{(i)}\right\}_{i=1}^{N}$ with probabilities $\left\{w_{t-1}^{(i)}\right\}_{i=1}^{N}$ to give $\left\{\tilde{\mathbf{x}}_{t-1}^{(i)}, \frac{1}{N}, \right\}_{i=1}^{N}$
      ii. Propagate weights $\tilde{w}_t^{(i)} = \frac{p(y_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)})}{\pi(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, y_{0:t})}$
   (b) For $i = 1, \ldots, N$
      i. Normalise weights $w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}$

Although resampling reduces the problem of degeneracy, it introduces other problems which did not exist with the SIS algorithm. Sample impoverishment happens when there is a loss of diversity in the particles; only a few of the particles have significant weight and the rest are insignificant. A common approach is to only resampling when it is needed. To detect when it is needed, the variability of the weights is monitored. This is commonly done by calculating the effective sample size (ESS) – which is N if the weights are all equal, and 1 if there is just a single non-zero weight. Thus often resampling is only performed when this ESS is lower than some pre-chosen threshold. Various strategies have been proposed to improve the performance of SMC methods in terms of accuracy, convergence, computational speed, etc. Surveys of these methods have been given over the years (Doucet et al., 2000; Cappé et al., 2007; Doucet and Johansen, 2009). Additionally, there is a wide variety of convergence results for SMC methods, most of which are concerned with the accuracy of the particle approximation of the distribution of interest as a function of N. For applications in state spaces, there are several studies on the stability of SMC as the time step grows

(Chopin, 2004; Douc and Moulines, 2007).

Sequential Monte Carlo in Graphical Models

Whilst the majority of approximate inference methods developed for graphical models are either variational methods or MCMC methods, recently there has been interest in how SMC can be used for the problem of inference in graphical models. Typically, SMC struggles in high dimensional problems (Rebeschini et al., 2015) and hence is not considered applicable to most graphical models. The first approaches that utilised SMC for graphical models use SMC within a loopy belief propagation framework (Isard, 2003; Sudderth et al., 2010). Even in the limit of the Monte Carlo sample size going to infinity, these methods are still approximate as they are taking approximations over loopy belief propagation.

As the dimension increases, the approximation to the target distribution quickly deteriorates when a single IS step is used (Bengtsson et al., 2008). The approximation error generally grows exponentially in the model dimension. However, this degeneracy can be avoided by starting off with a simple distribution and moving to the one of interest (Chopin, 2002). The stability of these problems is less well studied, although Beskos et al. (2014a) gives analytical understanding about the effect of dimension on SMC methods for static problems. The majority of SMC methods on graphical models work by slowly increasing the complexity of the graphical model to give a Monte Carlo sample which eventually approximates the full distribution for a static problem. For example, hot coupling (Hamze and de Freitas, 2005) starts with Monte Carlo approximation to a spanning tree of the MRF and edges are slowly added according to an annealing scheme. The method by Naesseth et al. (2014) works with the factor graph and sequentially adds factors to increase the probability space. As a byproduct of the algorithm, this also gives an unbiased estimate of the normalising constant. Both of these methods can be applied to discrete random variables and can be used within particle MCMC methods for parameter inference (Naesseth et al., 2014; Everitt, 2012).

Another problem that has recently been studied is SMC for inference in non-static problems

on networks. Methods include those which break down the graphical model into smaller blocks. The SMC is independently performed on lower dimensional blocks before correcting for the blocking (Rebeschini et al., 2015; Briggs et al., 2013). By breaking the graphical model down, these methods are only approximations to the state space. Beskos et al. (2014b) introduces a space-time particle filter which uses a local particle filter in space and a global particle filter in time. Similarly, nested SMC (Naesseth et al., 2016) takes an exact approximation (Andrieu et al., 2010) to the weights in the SMC algorithm. The proposal distribution is the one used in fully adapted SMC. Instead of computing this proposal exactly, the weights are approximated by running an internal SMC sampler for each weight on the state space model, hence producing an unbiased estimate of the weights to move from time t to time $t + 1$.

2.3.3 Bayes Linear Methodology

Variational Methods and Monte Carlo methods depend on a fully specified prior distribution for the latent variables. For large complex data, it is often difficult to fully specify a prior distribution, which agrees with experts' opinions (Craig et al., 1998). It may only be possible to specify partial prior beliefs. Instead of working with a full prior probability distribution, an alternative is to use Bayes Linear (BL) methodology. This methodology works with the expectations directly rather than probabilities. Beliefs about the latent random variables are expressed through expectations and covariances rather than full probability distributions (Goldstein and Wooff, 2007) and these values are updated given observations. No assumptions are made about the underlying probability distribution of the model. The method of using subjective expectation to quantify expectations and uncertainty is sometimes known as prevision. A rigorous development of the methodology for working directly with expectation as the primitive quantity is given in De Finetti (1974).

The link between subjective probability and expectations is easily justified. The expectations can be used to calculate if an event takes place using indicator functions (Whittle, 2012). If the indicator is one when the event takes place and is zero otherwise, the proba-

bility of the event is given by the expectation of the indicator function.

BL methodology allows beliefs to be specified at the level of detail required through a set of random quantities, $C = \{X_1, X_2, \ldots\}$ where for each $X_i, X_j \in C$, the prior beliefs are specified though:

- The expectation: $E(X_i)$.
- The variance $\mathrm{Var}(X_i)$ quantifying uncertainty in the judgement of $E(X_i)$.
- The covariance $\mathrm{Cov}(X_i, X_j)$ expressing the judgement of the relationship between the random variables, quantifying the extent to which X_i may linearly influence X_j.

For example, for two random variables X and Y, interest may lie in the set of random quantities $C = \{X, Y, X^2, Y^2, XY\}$. For each of these random quantities a prior expectation, along with variances and covariance would need to be specified to use BL methodology. An alternative way to specify a full probability distribution would be to specify the expectation of every possible combination of the random variables.

Formally, the belief specification is formed by constructing a linear space $\langle C \rangle$ which consists of all finite linear combinations of the elements of C. Covariance defines an inner product and norm over the linear space $\langle C \rangle$. Hence, the name Bayes Linear follows from this linearity underlying the inner product structure. Operations which can be done on linear subspaces can be done in BL methodology. Familiar properties, present when working with probabilities, are also present when working within BL methodology such as linearity, convexity, scaling and additivity.

Supposing a set of the random quantities are observed. The beliefs on the remaining expectations and variances are adjusted directly to incorporate the outcome. If the base of the analysis is $\mathbf{C} = \{\mathbf{X}, \mathbf{Y}\}$ and the n-dimensional set $\mathbf{Y}$ is observed, then judgement on $\mathbf{X}$ is adjusted given $\mathbf{Y} = \mathbf{y}$. The adjusted expectation and covariance are the best estimate for each X_i as a linear combination of the observed quantities. The best estimate is defined in

terms of minimising the mean squared error. Let $Y_0 = 1$, then the adjusted expectation is:

$$\widehat{E}\left[X_k|\mathbf{Y}\right] = \sum_{i=0}^{n} h_i Y_i, \tag{2.3.25}$$

which minimises:

$$E\left[\left(X_k - \sum_{i=0}^{n} h_i Y_i\right)^2\right], \tag{2.3.26}$$

over all collections of $\mathbf{h} = (h_0, h_1, \ldots, h_n)$ (Goldstein and Wooff, 2007).

This is a convex optimisation problem, and could be found using optimisation software. However, it is also possible to derive analytical equations for the best estimate. For any choice $\mathbf{h}_* = (h_1, \ldots, h_n)$, the value of h_0 is found by differentiating (2.3.26) with respect to h_0:

$$\begin{aligned}\frac{\partial}{\partial h_0}\left(E\left[\left(X_k - \mathbf{h}_*^T\mathbf{Y} - h_0\right)^2\right]\right) &= \frac{\partial}{\partial h_0}\left(E\left[X_k^2\right] - 2\mathbf{h}_*^T E\left[X_k\mathbf{Y}\right] - 2h_0 E\left[X_k\right] + 2\mathbf{h}_*^T E\left[\mathbf{Y}\right]h_0\right. \\ &\qquad \left. + \mathbf{h}_*^T E\left[\mathbf{Y}^T\mathbf{Y}\right]\mathbf{h}_* + h_0^2\right), \\ &= -2E\left[X_k\right] + 2\mathbf{h}_*^T E\left[\mathbf{Y}\right] + 2h_0. \end{aligned} \tag{2.3.27}$$

Then setting equation (2.3.27) equal to zero:

$$h_0 = E\left[X_k\right] - \mathbf{h}_*^T E\left[\mathbf{Y}\right]. \tag{2.3.28}$$

For this value of h_0, the value of $\mathbf{h}_*$ is found by differentiating (2.3.26) with respect to $\mathbf{h}_*$:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{h}_*}\left(E\left[\left(X_k - \mathbf{h}_*^T\mathbf{Y} - E\left[X_k\right] + \mathbf{h}_*^T E\left[\mathbf{Y}\right]\right)^2\right]\right) &= \frac{\partial}{\partial \mathbf{h}_*}\left(\mathrm{Var}(X_k) + \mathbf{h}_*^T\mathrm{Var}(\mathbf{Y})\mathbf{h}_* - 2\mathbf{h}_*^T\mathrm{Cov}(\mathbf{Y}, X_k)\right), \\ &= 2\mathbf{h}_*^T\mathrm{Var}(\mathbf{Y}) - 2\mathrm{Cov}(X_k, \mathbf{Y}).\end{aligned}$$

Assuming that $\mathbf{Var}(\mathbf{Y})$ is of full rank, setting this equal to zero gives:

$$\mathbf{h}_*^T = \mathrm{Cov}(X_k, \mathbf{Y})\mathrm{Var}(\mathbf{Y})^{-1},$$

and substituting into equation (2.3.25), the BL adjusted expectation is:

$$\widehat{E}\left[X_k|\mathbf{Y}\right] = E\left[X_k\right] + \mathrm{Cov}(X_k, \mathbf{Y})\mathrm{Var}(\mathbf{Y})^{-1}(\mathbf{y} - E\left[\mathbf{Y}\right]).$$

The adjusted variance of X_k given $\mathbf{Y}$ is:

$$\widehat{\mathrm{Var}}(X_k|\mathbf{Y}) = E\left[(X_k - \widehat{E}\left[X_k|\mathbf{Y}\right])^2\right].$$

Substituting in $\widehat{E}(X_k|\mathbf{Y})$ from (2.3.29), the adjusted variance is:

$$\widehat{\mathrm{Var}}(X_k|\mathbf{Y}) = \mathrm{Var}(X_k) - \mathrm{Cov}(X_k, \mathbf{Y})\mathrm{Var}(\mathbf{Y})^{-1}\mathrm{Cov}(\mathbf{Y}, X_k). \tag{2.3.29}$$

The adjusted expectations and variances can be viewed as a primitive which quantifies aspects of the beliefs given observations. Alternatively they can be seen as a simple tractable approximation to a fully Bayesian analysis for complicated problems. For linear Gaussian models, the adjusted values are equivalent to the posterior mean and covariance found using a fully Bayesian analysis.

A common application for BL methodology is for uncertainty analysis in high dimensional complex computer models for physical processes such as climate models (Williamson et al., 2013), the Thermohaline circulation in the Atlantic Ocean (Goldstein and Rougier, 2009) and galaxy formation (Vernon et al., 2010). Within these applications, the aim is to seek out areas of simulator inputs which closely match the historical observations (termed history matching in the oil industry). An emulator based on BL methodology can be used to make a decision where the simulator should be run in order to eliminate areas of uncertainity from the model. Many of the applications of BL aim to help with decision making within the application, such as deciding where to sink further wells in off-shore oil rigs (Craig et al., 1997) and using a decision theory based approach to experimental design (Farrow and Goldstein, 2006).

Chapter 3

Sequential Decision Problems

3.1 Introduction

In many real-world situations, decisions are made in order to help maximise some type of reward. The decisions are often difficult to make because of uncertainty in the system and about the outcome of the decision and any possible future outcomes. The decisions or the actions they generate do not only result in a reward but also result in new knowledge that can be used to help improve future decisions. This type of problem can be modelled as a sequential decision problem, which is a type of learning problem where the learning can be either offline or online (Powell and Ryzhov, 2012). In offline learning problems, no rewards are collected as the information is gathered. The decisions and feedback are sequential but a reward is only received at the end of the simulation. Ranking and selection is an example of an offline learning problem, where the concern is to find the best of two or more possible processes after a number of observations. See Kim and Nelson (2006) for an introduction to ranking and selection problems.

In online learning problems, the rewards are collected at the same time as information is gathered. Multi-armed bandit problems are one example of an online learning problem. They are one of the best known and most commonly used type of sequential decision problem

and arise in many different disciplines including statistics, operations research, machine learning, computing, control theory and economics. One of the original motivations for studying the multi-armed bandit problems coming from statistics was the sequential design of experiments (Robbins, 1952).

Sequential decision problems have been extensively studied because of the large number of applications. Different disciplines vary in their approach to these types of problems, driven by differing motivating applications. Machine learning has recently been very active in the area of research for multi-armed bandits. For example, in website optimisation (Scott, 2010), where the observations are easy to observe and the outcome is almost instantaneous, they combine the ideas of sequential learning motivation (Robbins, 1952) with ideas from reinforcement learning; the process of learning about uncertain environments and observing the outcomes inspired by behavioural psychology (Sutton and Barto, 1998). Many of the applications within statistics and operations research have less instantaneous rewards and a higher cost associated with them. For example, deciding where to drill to find oil, or deciding which drug to give a patient in a clinical trial.

3.1.1 Markov Decision Processes

A common way to model a sequential decision problem is as a Markov Decision Process (MDP), a discrete time stochastic process which is an extension of a Markov chain with the addition of actions and rewards. A detailed introduction to MDP is given in Whittle (1982) and Puterman (2014). The process moves through the states with some transition probability, which depends on the actions of the decision maker. Hence, the transition probability depends on the current state of the system as well as the action. Given the current state and action, the next state is conditionally independent of all previous states and actions. Hence, the state transitions satisfy the Markov property.

The problem in a MDP is to find a policy for the decision maker. A *policy* is a set of rules to decide which action is taken in which state. The standard objective is to maximise the total expected reward over the time horizon which can either be finite or infinite. For infinite

time horizons, the rewards are usually discounted by a fixed factor so that the total reward is finite. An action will not only be determined by the possible immediate reward but also any possible future rewards. For example, an action may result in a poor immediate reward but lead to good future rewards. Hence, evaluating the current best action results in looking at all possible future actions.

The multi-armed bandit problem is an example of a one state MDP, where learning is added to the problem (Vermorel and Mohri, 2005). This occurs when transition probabilities are not fully known and can be learnt from the outcome of the actions.

3.1.2 Multi-armed Bandits

A typical example of a multi-armed bandit problem is the problem of a gambler at a row of slot machines (sometimes called one-armed bandits). The gambler must decide which machines to play, how many times to play them and in which order they should be played. The objective of the gambler is to maximise the sum of their rewards over the sequence of plays. More generally, in a classical multi-armed bandit problem, the decision maker faces K possible actions (also known as arms). Associated with each possible action are unknown parameters, $\theta_a \in \Theta$. At each time step, $t = 1, 2, \ldots$ an action $a_k \in \{1, \ldots, K\}$ is taken and the action $a_t = a$ corresponds to choosing actions a at time t. This results in an observed reward y_t, which is assumed to be drawn from the marginal distribution ν_a with parameters θ_a:

$$Y_t \sim \nu_a(\theta_a). \tag{3.1.1}$$

The expectation of the reward on arm a at time t is denoted μ_a:

$$E[Y_t] = \mu_a. \tag{3.1.2}$$

From a Bayesian point of view, the parameter values $(\theta_1, \ldots, \theta_K)$ are drawn from a prior distribution. The objective is to define a policy which maximises the cumulative expected

reward over the time horizon n:

$$E\left[\sum_{i=1}^{n} Y_t\right] \tag{3.1.3}$$

over both the realisations and the prior information. For an infinite time horizon, the reward is multiplied by a discount factor, γ, and the policy is chosen which maximises:

$$E\left[\sum_{i=1}^{\infty} \gamma^t Y_t\right]. \tag{3.1.4}$$

In the past couple of decades, many algorithms have been suggested for determining the best policy for the multi-armed bandit problem. These problems can be seen as a type of exploration-exploitation problem. At each stage, there is a trade-off between sampling a reward on an arm which appears to be doing well based on the limited information of previous rewards (exploitation) and sampling a reward on an arm which may look inferior because of sampling variability (exploration) (Auer et al., 2002). A simple heuristic policy is the greedy policy where at each time step, the action with the highest posterior expected reward, $\mu_{j,t}$, is picked:

$$a_t = \text{argmax}_{j=1,\ldots,K}\mu_{j,t}. \tag{3.1.5}$$

The method can be seen as a pure exploitation method. Better methods will combine both exploration and exploitation in order to maximise the future rewards. Section 3.2 gives details of some alternative policies.

3.2 Solutions

Making a decision in multi-armed bandit problems involves considering the outcome of all future possible decisions which can quickly become unmanageable. An alternative is to use stochastic dynamic programming (Bellman, 1954) approaches which break the problem down into smaller, more manageable problems. Using dynamic programming, the optimal

solution to the multi-armed bandit problem reduces to an index problem. An *index* is a numerical value independently assigned to each of the arms and the policy is chosen by selecting the arm with the highest index. Gittins and Jones (1979) proved that the optimal policy for the infinite horizon multi-armed bandit problem, with discounted rewards where the rewards are conditionally independent given the action, is an index policy. They termed the index the dynamic allocation index but it is now commonly referred to as the Gittins index. Easier to follow proofs have since been given to show that Gittins index is optimal for multi-armed bandit problems (Tsitsiklis, 1994). One way the Gittins index can be derived is by considering a smaller independent problems for each arm where the arm is pulled and it must be decided adaptively when to stop pulling that arm. Upon stopping a fixed retirement cost is given. The Gittins index is the arm with the largest fixed retirement cost required to stop pulling the arm now (Frazier, 2011). The solution is reduced from solving a k-armed bandit problem to solving the Gittins indices for k one-armed bandits. For many problems these optimal policies are hard to compute, and for finite horizon problems Gittins indices are no longer optimal; the proof is fragile to changes in the assumptions. As a result, research has turned to heuristic methods which are not guaranteed to select the optimal policy at each time step to solve multi-armed bandit problems.

An alternative way to measure how well a decision policy performs is to minimise the cumulative regret (Robbins, 1952). If μ^* is the maximum expected reward over the K arms then instead of maximising the cumulative reward, the multi-armed bandit problem can be thought of as minimising the cumulative regret:

$$R_n(\theta) = \sum_{t=1}^{n} (\mu^* - \mu_{a_t}). \tag{3.2.1}$$

A basic objective given by Robbins (1952) is that $R_n(\theta)/n \to 0$ as $n \to \infty$. Policies which achieve this objective are said to be consistent (Lai and Robbins, 1985). Stronger conditions consider the worst case of regret. Lai and Robbins (1985) show that regret must grow at least logarithmically with time and provide a lower bound on the number of suboptimal draws in a good strategy for a single armed bandit. These are extended to multi-armed bandits

by Burnetas and Katehakis (1996). As well as showing that policies asymptotically reach this lower bound, the current standard in evaluating policies is to give an upper bound of the regret in finite time (Auer et al., 2002). The aim is to show that the regret is of $O(\log(n))$ but large constant multipliers mean this is quite a loose condition. Approaches to finding policies include upper confidence bound methods, interval estimation and Thompson sampling. In the following sections, different approaches to heuristic policies are discussed. These policies are heuristic as they do not guarantee selecting the optimal policy at each time step, unlike the Gittins index. However, they can be applied to a far wider range of problems, for which no optimal policy exists, and have been shown in many cases to reach the lower bound on the number of sub optimal draws. There are also many more general heuristic methods which can be applied to a broader class of optimal learning algorithms (Vermorel and Mohri, 2005; Luce, 2005).

3.2.1 Semi-uniform Methods

Multi-armed bandit problems are an exploration-exploitation problem. Methods which distinguish between exploration steps and exploitation steps are called semi-uniform methods. The simplest and most widely used semi-uniform method is the ϵ-greedy policy (Watkins, 1989). The ϵ-greedy policy randomly chooses an arm with probability ϵ and otherwise makes a greedy choice, selecting the arm with the highest expected reward:

$$a_t = \begin{cases} \max_{j=1,\ldots,K} \mu_{j,t} & \text{with probability } 1-\epsilon \\ \text{random choice} & \text{with probability } \epsilon \end{cases} . \tag{3.2.2}$$

The choice of $\epsilon \in (0,1)$ is left to the user; a smaller value of ϵ results in less exploration. The ϵ-greedy policy is a sub-optimal policy because the constant factor ϵ prevents the strategy from getting arbitrarily close to the optimal action (Vermorel and Mohri, 2005). Variants of the ϵ-greedy policy have been developed, such as the ϵ-decreasing strategy which consists of using a decreasing ϵ getting arbitrarily close to the optimal strategy asymptotically. Auer et al. (2002) shows that, under certain constraints on how ϵ decreases, this achieves an

$O(log(T))$ regret.

Semi-uniform strategies are simple heuristic methods. Methods which assign a probability to selecting each possible arm may perform better. These methods give arms with higher expected rewards a higher probability of being chosen but incorporate exploration by still giving small probabilities to those arms with lower expected reward. One such method is the Softmax algorithm (Vermorel and Mohri, 2005) which assigns a Boltzmann distribution to the probability of selecting each arm:

$$\frac{e^{\mu_{k,t}\tau}}{\sum_{i=1}^{K} e^{\mu_{i,t}\tau}} \tag{3.2.3}$$

where τ is the user defined tuning parameter where a larger value of τ results in more exploration.

3.2.2 Optimistic Methods

A different approach to incorporating exploration is used in optimistic methods which take an optimistic guess of the mean reward in the face of uncertainty. The reward with the highest optimistic estimate is then chosen. The methods prefer actions which have a high amount of uncertainty. For two arms with similar expected rewards, the arm which has a higher amount of uncertainty about that expectation will have a larger optimistic reward. If the optimistic view of the expected reward is wrong then after choosing that action a few times, the value will quickly decrease and it will no longer be chosen. These methods are intuitive as they are an approximation to the type of behaviour on Gittins indices which can be seen as the mean reward plus some type of uncertainty bonus.

One optimistic approach is called Interval Estimation (Kaelbling, 1993). Associated with each arm is a $100(1-\alpha)\%$ upper bound on the mean reward, where α is a parameter in the range $(0, 1)$. The value of α is left to the user; a smaller value of α leads to more exploration. At each round, the action of highest upper bound is chosen. For example, if a reward on arm a at time t is normally distributed with posterior mean, $\mu_{a,t}$ and posterior standard

deviation $\sigma_{a,t}$. Interval estimation chooses the policy:

$$a_t = \text{argmax}_{j=1,\ldots,K} \mu_{i,t} + z_\alpha \sigma_{i,t} \tag{3.2.4}$$

where z_α is the is the $1-\alpha$ quantile of the standard normal distribution.

Alternatively, Upper Confidence Bound (UCB) methods have become popular optimistic heuristic methods for solving the multi-armed bandit problem. They are based on an upper confidence bound to the mean reward on each arm and are designed to meet the regret optimality criteria. Many of the methods do not make any assumptions on the probabilistic model and rely only on the sample mean of rewards on the arms. For each arm, an index is calculated based on an upper bound on the expected reward, which is a function of the number of times the arm has been pulled up to the current time. Many different UCB policies have been suggested and designed for specific problems (Auer et al., 2002; Garivier and Cappé, 2011). A general Bayes-UCB policy has been proposed by Kaufmann et al. (2012a) which is inspired by a Bayesian perspective to the standard (frequentist) UCB policies. The Bayes-UCB policy chooses the arm which has the highest value of some user-chosen quantile, α_t, of the posterior distribution of the rewards. More specifically, denote the quantile function associated to the distribution ν as $Q(\lambda, \nu)$, such that $P(X \leq Q(\lambda, \nu)) = 1$ and the posterior distribution for arm a at time t as $\nu_{j,t}$. Bayes-UCB chooses the policy:

$$a_t = \text{argmax}_{j=1,\ldots,K} Q(1-\alpha_t, \nu_{j,t}) \tag{3.2.5}$$

In many cases, the method can reach the lower bound on regret discussed in Section 3.2.

3.2.3 Randomised Probability Matching

A naturally Bayesian approach to defining a policy is to use randomised probability matching. The algorithm works by sampling from the posterior distribution on each arms and choosing the arm with the highest value. A sample, $\tilde{\theta}_a$, is drawn from the posterior distri-

bution $\nu_{a,t}(\theta_a \mid \cdot)$ on each arm and the arm selected is:

$$a_t = \text{argmax}_{j=1,\ldots,K}\tilde{\theta}_a. \tag{3.2.6}$$

Thompson sampling (Thompson, 1933) was the first example, and was proposed as a method for solving stochastic bandits with Bernoulli rewards. Recently, the method has been shown to be asymptotically optimal for cumulative regret minimisation (Kaufmann et al., 2012b). Although proposed for Bernoulli bandits, the method can easily be extended to general stochastic bandit problems (Agrawal and Goyal, 2012). An optimistic version of this has been developed which instead takes the maximum of the mean posterior reward and the sample (May et al., 2012):

$$a_t = \text{argmax}_{j=1,\ldots,K} max(\tilde{\theta}_a, \mu_{a,t}). \tag{3.2.7}$$

3.3 Multi-armed Bandit Problems on Networks

Many variants of the multi-armed bandit problem have been proposed, where the variations can be a result of a change in the reward, the available actions or the feedback. One variant of the classic multi-armed bandit problem is when there is some underlying dependence structure between the rewards: the arms are correlated. The assumptions behind the Gittins index do not hold for this problem because statistical dependence among the arms means the indices can not be decomposed between the arms. When one arm is pulled, knowledge of other arms in the network is also gained. Methods which involve specific correlation structures include: the Knowledge Gradient Policy (Ryzhov et al., 2012) which can handle correlated Gaussian rewards; dependencies among bandits in the form of clusters (Pandey et al., 2007); and linearly parameterised arms (Rusmevichientong and Tsitsiklis, 2010). The dependent bandit problem is also related to bandit problems with side observations (Wang et al., 2005) where there is a separate process that provides additional information about the reward process at each time point; no such separate information about the reward process is present in the dependent bandit.

Dependencies in the multi-armed bandit problem can also be in the form of a graph structure to the bandits. Many multi-armed bandit problems on graphs are concerned with sequential decision problems with limited feedback (Caron and Bhagat, 2013; Cesa-Bianchi et al., 2013). For example, finding the most influential person in a social network to target for marketing purposes. This type of application is becoming more common as the use of social media plays a larger role in marketing. The methods tend to work by information on an arm being shared with a neigbouring node in the network, resulting in local influence (Carpentier and Valko, 2016).

Dimitrov et al. (2016) model a communication network as a MRF to help efficiently find intelligence items relevant to a query. It is assumed that if a person is involved in a relevant conversation with one person, they are more likely to be involved in relevant conversations with other people too. The MRF is used to model the dependence structure between individuals, and observations are made on edges in the network. Sequential decision problems on networks were also used for exploring an oil and gas field in the North Sea (Brown and Smith, 2013; Martinelli et al., 2013). A natural approach for dealing with sequential decision making in graphical models in general is given by Brown and Smith (2013), who propose splitting the network up into several disjoint clusters and using dynamic programming on the clusters. Approximation techniques are then used to find a policy in the full network. The results rely heavily on bandit superprocesses, a generalisation of multi-armed bandits and focus on computing optimal policies for the exploration-exploitation problem.

Chapter 4

Bayes Linear Methods for Large-Scale Network Search

4.1 Introduction

There are many applications where you wish to search through a large set of items, such as emails or documents to find a small set of them which are relevant to a query. In Chapter 1 we introduced the idea of network-based searches for finding relevant items distributed across the edges of a network. We give a quick summary of two specific motivating examples, which we return to later in the chapter.

The first example comes from corporate law suits. Companies are often required to hand over large databases of emails to one another and the information within these emails may make up part of their legal battle (Cavaliere et al., 2005). To find and judge which of the emails are relevant to a case requires searching through a huge network of emails, the majority of which will be irrelevant, in possibly time pressured situations, at the cost of millions of dollars and thousands of hours (Flynn and Kahn, 2003).

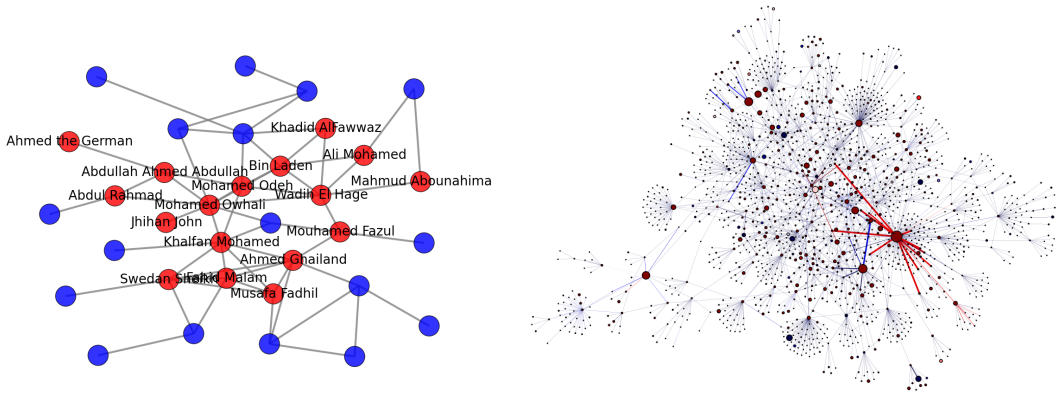
The second example comes from intelligence processing. Efficient military operations and law enforcement rely on the timely processing of intelligence (Hughbank and Githens, 2010).

Overwhelming amounts of intelligence is collected daily, particularly communications intelligence where the use of social media, text messaging and emails have drastically increased (Duyvesteyn et al., 2014). A result of the overwhelming amount of intelligence collected is that often the processor is faced with a bottle neck of intelligence items: far more are collected than can be processed. In a time critical situation, the main challenge for the processor is to provide the analysts with the largest set of relevant intelligence items in the given time window.

In these examples, we have a set of emails or other communications, henceforth called *items* for simplicity, and we wish to search this set of items to find those which are relevant to the query of interest. Each of the items involves two or more participants. The participants and the items between them induce a network where the participants form the nodes. An edge exists between two nodes if there is at least one item between the two associated participants. Participants can also be relevant or irrelevant to the query. Relevant items are more likely to occur between relevant participants. In addition, it is likely that relevant participants will cluster in groups. Thus, the network contains information that can be exploited to help decide which items to observe. Furthermore, each of the problems is time pressured and could involve huge amounts of possible items to search through. Hence, they require scalable search methods in order to be able to process as many items as possible in a short period of time and select items to observe which are likely to give as many relevant items as possible.

The main aim of the network based search is to find the relevant items on the edges, and hence learn on which edges you are most likely to find a relevant item. Throughout this thesis, the algorithms focus on this aim. However, as a side product of the network based search we also learn about the relevance of the participants in the network – which, in itself, is useful information in the examples given above.

Two example network related to the motivating applications above are shown in Figure 4.1.1. The first is based on a possible terrorist network associated with the bombing of the US embassy in Tanzania in 1998. The second is constructed from Enron email data. The



(a) Tanzania network

(b) Network constructed from Enron email data

Figure 4.1.1: Two example networks. Figure 4.1.1a shows a possible terrorist network associated with the bombing of the US embassy in Tanzania in 1998. Figure 4.1.1b shows a network constructed from Enron email data.

Enron Corpus consists of hundreds of thousands of emails from roughly 150 senior employees at the Enron Corporation. The dataset was made public during the US government's legal investigation of the Enron Corporation after its collapse in 2001 and has similarities with the datasets of corporate legal cases. The width of the edge or node corresponds to the total number of items associated with it and the more red (less blue) an edge or node is, the greater the number of relevant items associated with it. It can be see that the relevant edges/nodes tend to cluster together in the network. An efficient method for network-based search will need to utilise this clustering property.

The main contribution of this chapter is to develop an approximate inference method within a binary MRF to allow for efficient searching. We take a sequential decision approach to the problem, introduced by Dimitrov et al. (2016), and develop an approximate inference method for sequential inference on binary networks. The method is based on BL methodology; we add additional constraints to the optimisation form of the BL approximation so that it produces results that make sense for the application under consideration. For network-based searches we show how these constrained BL updates can be used and that they perform well, in the context of communication networks, compared to a model where the network is not considered.

4.1.1 Problem Setting and Related Work

We use a class of models for the relevance of participants and items that Dimitrov et al. (2016) introduce for intelligence processing. Under this model, each participant is either relevant to the query or irrelevant, modelled through a set of binary but unknown random variables, one for each participant. In addition, a random variable is associated with each edge describing the probability of observing a relevant item between the two associated participants. This random variable representing the probability of finding relevant items on an edge is dependent on the involved participants' relevance values. An item involving two relevant participants is more likely to be relevant to the query than at item with at least one participant who is irrelevant.

We call the person who searches through the items the *user* and assume the user correctly identifies which items are relevant; there are no false positives or false negatives. The users task is to identify as many relevant items as possible, where there are far more items than can be observed during a limited available time period. Items are observed one at a time, with the user gaining information that can be exploited to focus the future search more effectively.

The problem has three related steps:

1. Constructing an appropriate joint prior distribution for the relevance of participants and items.
2. Updating this joint distribution as items are observed.
3. Deciding which item to observe next, given the current joint distribution on the relevance of items and participants.

Dimitrov et al. (2016) show that the success of the user finding relevant items can be substantially improved by using a prior distribution for the relevance of participants which models the fact that relevant participants cluster together. Their prior distribution is based upon classifying each participant as either relevant to the query or irrelevant, modelled

through a set of unobserved binary random variables, one for each participant. The joint distribution of these binary random variables is specified as a MRF. This MRF introduces local dependencies that encourage relevant participants to cluster together. In addition, associated with each edge is a random variable describing the probability of observing a relevant item between the two associated participants. These random probabilities are dependent on the involved participants' relevance values. An item involving two relevant participants is more likely to be relevant to the query than one where at least one participant is irrelevant.

The limitation of the model suggested by Dimitrov et al. (2016) is that updating the joint distribution in step (2) can be computationally prohibitive for large networks. Using an exact inference method such as variable elimination (Zhang and Poole, 1994), conditioning (Shachter et al., 1994) or junction trees (Lauritzen and Spiegelhalter, 1988), the computational cost grows exponentially in the tree width of the graph. Hence, they are intractable for many large networks (Koller and Friedman, 2009). Ellis (2013) demonstrates that the use of exact inference can limit the size of network for which the method is computationally tractable to a few hundred nodes. Thus while the approach of Dimitrov et al. (2016) can be used on the Tanzania network in Figure 4.1.1a, it is not practicable for larger networks such as that constructed from the Enron data in Figure 4.1.1b. For such larger networks, approximate inference is required to overcome the computational intractability of exact inference.

There are a number of standard heuristic policies for performing the third step. Importantly for our work, most of these policies depend on the posterior distribution for the probabilities that items are relevant just through the posterior mean and covariance. This motivates using the BL methodology (Goldstein and Wooff, 2007), which summarises the prior and posterior distribution just through their means and covariances and gives a simple procedure to update these given new data. Thus, the use of BL has many advantages for this application. Firstly, it simplifies specifying the prior, as we need only specify the prior mean and covariance. Secondly, the BL updates are computationally practical for large networks; with their worst case computational cost scaling as the cube of the number of

nodes in the network, compared to exponentially for the exact Bayesian updates. Finally, whilst the BL updates are approximate, they are approximations that focus on the aspects of the posterior, namely the mean and covariance, that are needed for the decision problem in step (3).

Despite these advantages, the standard BL updates are inappropriate for our application, as they ignore the fact that we have a posterior distribution on parameters that represent binary random variables and thus the posterior mean of these parameters will lie in the interval $[0, 1]$. The standard BL updates do not force the posterior expectation to lie inside this range. We introduce an extension of the BL update that respects this constraint.

The outline of the chapter is as follows. In Section 4.2, we introduce the model of Dimitrov et al. (2016). Section 4.3 gives an overview of the BL methodology. In Section 4.3.1 the constrained BL optimisation problem is introduced, where additional constraints are added to the BL optimisation problem to ensure the BL posterior means remain in the range $[0, 1]$ for any probabilities. We show the constrained BL updates have an analytical solution, allowing for savings in the computational cost of inference compared with using the constrained optimisation problem directly. Section 4.4 describes how the BL updates are used for approximate inference within the network-based search method. We evaluate the accuracy of the BL procedure both for approximating the posterior distribution of the MRF model, and for the network-based search problem in Section 4.5. These results include analysis of data based on a terrorist network and data taken from the Enron Corpus. In particular we show how BL can perform network-based search for large-scale networks, and that it leads to substantially improved performance over methods that ignore the dependence structure implied by the network.

4.2 The Model

The model we use for intelligence collection is originally described by Nevo (2011). Let $G = (V, E)$ denote the graph associated with the network. The nodes in the graph, V,

represent the participants. The edge $(u, v) \in E$ exists if and only if there is at least one item between participants u and v.

Let there be m nodes in the graph and n edges. Associated with each node is a random variable describing how relevant the participant is to the query. Define this random variable as Z_u for $u \in 1, \ldots, m$ and the set of participants' relevance values as $\mathbf{Z} = \{Z_1, \ldots, Z_m\}$. We assume that there is dependence between the random variables. A participant is more likely to be relevant if they share items with a relevant person and vice versa. We model the relevance of participant u as a binary random variable where:

$$Z_u = \begin{cases} 1 & \text{if participant } u \text{ relevant to query} \\ 0 & \text{otherwise} \end{cases}.$$

A random variable P_{uv} for edge $(u, v) \in E$ describes the probability of observing a relevant item on that edge. Let the set of probabilities be $\mathbf{P} = \{P_{uv}, (u, v) \in E\}$. We assume that the distribution of each P_{uv} will depend on Z_u and Z_v (the participants' relevance values). Given the involved participants' relevance values, the probabilities are conditionally independent. For conjugacy, we model these prior conditional probabilities using a beta distribution,

$$P_{uv} \mid Z_u, Z_v \sim \text{Beta}(a(Z_u, Z_v), b(Z_u, Z_v)). \tag{4.2.1}$$

The joint distribution over all random variables is:

$$P(\mathbf{Z}, \mathbf{P}) = P(Z_1, \ldots, Z_m) \prod_{(u,v)\in E} P(P_{uv} \mid Z_u, Z_v). \tag{4.2.2}$$

Both the participants' relevance values and the probabilities are latent random variables. They are not observed directly but learned by observing items on edges. When an item is observed, it is either relevant or irrelevant to the query. Suppose the kth item is observed on edge (u, v), and Y_{uv}^k denotes the observed binary relevance value of the item. That relevance value is modelled by the Bernoulli distribution with success probability P_{uv}. After n_{uv} observations on edge (u, v), the outcome of the observed items is stored through the

sufficient statistics n_{uv} and Y_{uv} where,

$$Y_{uv} = \sum_{k=1}^{n_{uv}} Y_{uv}^k, \tag{4.2.3}$$

is the number of relevant items observed. The joint posterior distribution over the participants' relevance values and the probabilities that edges produce relevant items, given the sufficient statistic Y_{uv} on each edge, $\mathbf{Y} = \{Y_{uv}, (u,v) \in E\}$ is:

$$\begin{aligned} P(\mathbf{Z}, \mathbf{P} \mid \mathbf{Y}) &= P(\mathbf{Z} \mid \mathbf{Y}) P(\mathbf{P} \mid \mathbf{Z}, \mathbf{Y}), \\ &= P(\mathbf{Z} \mid \mathbf{Y}) \left[\prod_{(u,v) \in E} P(P_{uv} \mid Z_u, Z_v, Y_{uv}) \right]. \end{aligned} \tag{4.2.4}$$

The equality above comes from the fact that given values of Z_u, Z_v, the variable P_{uv} is independent of other $P_{u'v'}$, $Y_{u'v'}$ variables. The conditional posterior for each P_{uv} will be

$$P_{uv} \mid Z_u, Z_v, Y_{uv} \sim \text{Beta}(a(Z_u, Z_v) + Y_{uv}, b(Z_u, Z_v) + n_{uv} - Y_{uv}).$$

Dimitrov et al. (2016) complete this model by specifying an MRF for $\mathbf{Z}$. Such a model introduces a local dependence structure with, for example, participants that share an edge in the network being more likely to be of the same type: either both relevant or both irrelevant.

Whilst a natural model, this leads to difficulties with evaluating this joint posterior distribution as calculating $P(\mathbf{Z} \mid \mathbf{Y})$ can be computationally prohibitive. Using exact inference algorithms the computational cost of evaluating this grows exponentially with the tree width of the graphical model in the worst case (Koller and Friedman, 2009). See Nevo (2011) and Ellis (2013) for more information on the exact updating process.

4.2.1 Example of Updates

To clarify the model, we present a simple example. The set of random variables for the participants' relevance values are given by $\mathbf{Z} = (Z_0, Z_1, Z_2)$ and the prior probabilities by

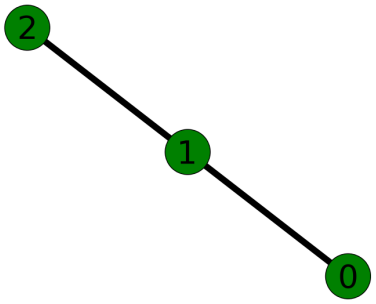


Figure 4.2.1: Network structure for simple example

Z_i	Z_j	$\phi(Z_i, Z_j)$
0	0	0.3
0	1	0.2
1	0	0.2
1	1	0.3

(a) Prior Clique Factor

Z_i	Z_j	$a(Z_i, Z_j)$	$b(Z_i, Z_j)$
0	0	1	4
0	1	1	2
1	0	1	2
1	1	1	1

(b) Prior $P_{ij} \mid Z_i, Z_j$ Parameters

Table 4.2.1: The prior distributions used for the example in Section 4.2.1 for the network in Figure 4.2.1. Table 4.2.1a is the prior clique factor over pairs of participants in the network which are connected by an edge and Table 4.2.1b is the conditional beta distribution parameters.

$\mathbf{P} = (P_{01}, P_{12})$. The corresponding network is shown in Figure 4.2.1. To specify a prior distribution, we give a factor over pairs of nodes in the network, say Z_i and Z_j, displayed in Table 4.2.1a. In other words, this factor appears twice in the joint distribution, once for each pair of connected nodes. Using the Hammersley-Clifford Theorem (Hammersley and Clifford, 1971), the joint distribution of $\mathbf{Z}$, can be calculated from these factors. The prior parameters for the dependant beta distributions are given in Table 4.2.1b. The data in Table 4.2.1 completely specify the prior joint distribution through equation (4.2.2).

After an observation, the posterior distribution for $\mathbf{Z} \mid \mathbf{Y}$ can be found using an exact inference method. An example of the change in beliefs over the network is given in Figure 4.2.2. The area of the node is proportional to its marginal posterior expectations and the thickness of the edges indicates the marginal posterior probabilities of observing relevant items on each edge. In addition, we display the exact posterior expectations.

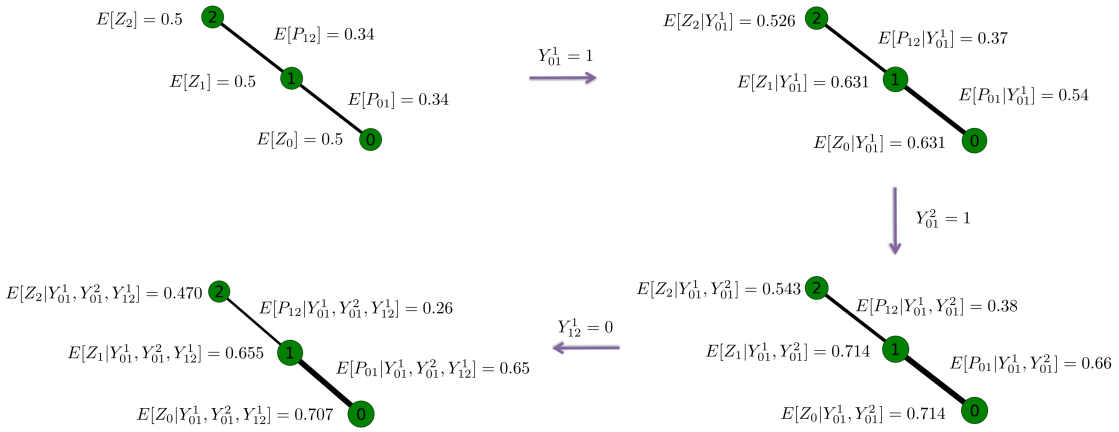


Figure 4.2.2: The updated expectations for the participants' relevance values and updated probabilities on each edge given the set of observations $\mathbf{Y} = (Y_{01}^1 + Y_{01}^2, Y_{12}^1) = (2, 0)$, for the simple line network. The prior distribution is given by the prior clique distribution in Figure 4.2.1a and the prior conditional beta distribution in Figure 4.2.1b. The area of a node represents the associated participants' expected relevance value given observations $E[Z_i \mid \mathbf{Y}]$ and the expected probabilities are given by the width of the edges. As relevant items are observed, the expected probability of observing a relevant item increases, as does the involved nodes relevancies. This information is propagated throughout rest of the network.

4.2.2 Sequential Decision Making

The reason we need to calculate the posterior distribution (4.2.4), is that given the current set of observations, $\mathbf{Y}$, we want to choose which item to observe next. We can define this as a Bayesian sequential decision problem, where we wish to maximise the number of relevant items observed over a fixed time interval, with the policy of which item to observe next depending on which of the items to date have been relevant. Solving this decision problem optimally is intractable, but there are many heuristic policies that have been shown to perform well (Auer et al., 2002).

A simple policy would be to observe the item which is currently considered most likely to be relevant. That is, we would choose an item from the edge (u, v) for which the posterior expectation of P_{uv} is highest. This is called the greedy policy. In practice, the greedy policy can often perform poorly, particularly for decision problems over a long time interval. It just exploits the current information as opposed to also trying to learn more about which edges have the highest P_{uv} values. As a result, there are more refined heuristic methods that take account of not just the posterior means of the $\mathbf{P}$ but also the posterior variances (Lai, 1987; Kaelbling, 1993; May et al., 2012). A high posterior variance means a large amount of uncertainty in the probability on an edge. Screening items on edges with high variances leads to greater learning about the actual probabilities associated with each edge compared to screening items on edges with low variances. The knowledge gained from such screenings can then be exploited in later decisions. Hence these policies preferentially screen items on edges with both high means and variances.

To implement one of these policies, we do not need to calculate the full posterior distribution for the P_{uv}s, but just the posterior mean and variance. This motivates the use of BL methods, which are based on Bayesian modelling and updates which solely use the mean and variance.

4.3 Bayes Linear Methodology

BL (Goldstein and Wooff, 2007) replaces the exact Bayesian update in (4.2.4) with an approximation. The idea of BL is to consider only the mean and covariance of the parameters. This simplifies the prior specification as only a mean and variance is needed, rather than the full distribution. On observing data, these are updated to produce approximations to the posterior mean and variance. For our application, BL requires specifying the prior expectation of the latent variables, $\mathbf{Z} = \{Z_1, \ldots, Z_m\}$, and the observable quantities $\mathbf{Y} = \{Y_1, \ldots, Y_n\}$. The uncertainty in the expectations of the random variables and the extent that one random variable will influence another is specified through their joint prior covariance,

$$\mathrm{Cov}(\mathbf{Z}, \mathbf{Y}) = \begin{pmatrix} \mathrm{Var}(\mathbf{Z}) & \mathrm{Cov}(\mathbf{Z}, \mathbf{Y}) \\ \mathrm{Cov}(\mathbf{Y}, \mathbf{Z}) & \mathrm{Var}(\mathbf{Y}) \end{pmatrix}.$$

The BL updates can be defined in terms of finding the best estimate of each Z_k by a linear combination of the data. This best estimate is defined in terms of minimising the mean squared error. The BL posterior mean for Z_k is just the resulting estimate and the BL posterior variance is defined as the variance of these estimators. Formally, for each $k \in \{1, \ldots, m\}$ we wish to find the coefficients $\hat{\mathbf{h}}^k = \left(\hat{h}_0^k, \hat{h}_1^k, \ldots, \hat{h}_n^k\right)$ that solve the following optimisation problem:

$$\underset{\mathbf{h}^k}{\text{minimise}}\ E\left[\left(Z_k - h_0^k - \sum_{i=1}^{n} h_i^k Y_i\right)^2\right], \quad k \in \{1, \ldots, m\}. \tag{4.3.1}$$

Then, for observed data $\mathbf{y} = (y_1, \ldots, y_n)$, we define the estimated posterior expectation as:

$$\widehat{E}\left[Z_k \mid \mathbf{Y}\right] = \hat{h}_0^k + \sum_{i=1}^{n} \hat{h}_i^k y_i, \tag{4.3.2}$$

for $k \in \{1, \ldots, m\}$ and the updated estimated posterior covariance between Z_k and Z_l as:

$$\widehat{\mathrm{Cov}}\left(Z_k, Z_l \mid \mathbf{Y}\right) = E\left[\left(Z_k - \hat{h}_0^k - \sum_{i=1}^{n} \hat{h}_i^k Y_i\right)\left(Z_l - \hat{h}_0^l - \sum_{j=1}^{n} \hat{h}_j^l Y_j\right)\right], \tag{4.3.3}$$

for $k, l \in \{1, \ldots, m\}$. The expectation in equation (4.3.1) is over both the latent variables and the observable quantities. By multiplying out the expectation, we can see that $\mathbf{h}^k$ depends on the prior specification of the expectation and covariance of $\mathbf{Z}$ and $\mathbf{Y}$. The optimisation problems defined by (4.3.1) are standard convex quadratic optimisation problems (Boyd and Vandenberghe, 2004) and can be solved analytically to give

$$\hat{\mathbf{h}}_{1:n}^k = \mathrm{Cov}(Z_k, \mathbf{Y})\mathrm{Var}(\mathbf{Y})^{-1}, \tag{4.3.4}$$

and

$$\hat{h}_0^k = E[Z_k] - (\hat{\mathbf{h}}_{1:n}^k)^T \mathbf{y}. \tag{4.3.5}$$

Thus the BL updated expectation is:

$$\widehat{E}\left[Z_k \mid \mathbf{Y}\right] = E\left[Z_k\right] + \mathrm{Cov}(Z_k, \mathbf{Y})\mathrm{Var}(\mathbf{Y})^{-1}(\mathbf{y} - E\left[\mathbf{Y}\right]), \tag{4.3.6}$$

and the BL updated covariance is:

$$\widehat{\mathrm{Cov}}\left(Z_k, Z_l \mid \mathbf{Y}\right) = \mathrm{Cov}(Z_k, Z_l) - \mathrm{Cov}(Z_k, \mathbf{Y})\mathrm{Var}(\mathbf{Y})^{-1}\mathrm{Cov}(\mathbf{Y}, Z_l). \tag{4.3.7}$$

For a full derivation of the update equations see Goldstein and Wooff (2007).

4.3.1 Constrained Bayes Linear

For a set of binary latent variables, we have the property that the true posterior expectation of the latent variable Z_u, $u \in \{1, \ldots, m\}$ will be in the range $[0, 1]$. Therefore, a desirable property of any approximation to the posterior expectation is that this still holds. The BL

updated expectation, (4.3.6), does not necessarily have this property.

To overcome this, we can recast the BL updates in terms of their original optimisation problem and modify (4.3.1) to include appropriate constraints on the posterior mean. In our case, the desirable property is that the updated expectation of the random variables remains in the range $[0, 1]$. This can be achieved by adding linear inequality constraints to the BL optimisation problem. The constrained form of BL updates, for binary random variables, is given by:

$$\begin{aligned} \underset{\mathbf{h}^k}{\text{minimise}} \quad & E\left[\left(Z_k - h_0^k - \sum_{i=1}^{n} h_i^k Y_i\right)^2\right], \\ \text{subject to} \quad & h_0^k + \sum_{i=1}^{n} h_i^k y_i \leq 1, \\ & h_0^k + \sum_{i=1}^{n} h_i^k y_i \geq 0, \end{aligned} \tag{4.3.8}$$

where, as above, $\mathbf{y} = (y_1, \ldots, y_n)$ are the observed quantities. The values needed for the constrained optimisation problem (4.3.8) are found by expanding the objective function

$$\begin{aligned} E\left[\left(Z_k - h_0^k - \sum_{i=1}^{n} h_i^k Y_i\right)^2\right] = E[Z_k^2] - 2h_0^k E[Z_k] - 2\sum_{i=1}^{n} h_i^k E[Z_k Y_i] + 2h_0^k \sum_{i=1}^{n} h_i^k E[Y_i] \\ + (h_0^k)^2 + \sum_{i=1}^{n}\sum_{j=1}^{n} h_i^k h_j^k E[Y_i Y_j]. \end{aligned} \tag{4.3.9}$$

To distinguish the constrained BL values for $\mathbf{h}$ from the unconstrained BL ones, we will denote them as $\tilde{\mathbf{h}}$. As in the case of unconstrained BL updates, the resulting optimisation problem is convex because the coefficients of the square terms, $h_i^k h_j^k$, make a positive semi-definite matrix. Because of this, the constrained optimisation problem can be solved using available convex optimisation software (Andersen et al., 2013). Constrained BL updates produce fundamentally different solutions than unconstrained BL updates. For example, for unconstrained BL updates, the $\hat{\mathbf{h}}^k$ do not depend on the actual value of the observations, just their expectation and covariances. On the other hand, when one of the constraints in (4.3.8) is binding, $\tilde{\mathbf{h}}^k$ will depend on the observations. Once $\tilde{\mathbf{h}}^k$ values are computed, BL

updated expectations and covariances are found using equations (4.3.2) and (4.3.3).

Solving problem (4.3.8) using a convex optimisation solver can be slow in practice. However, it is possible to analytically solve (4.3.8) using the following lemma.

Lemma 4.3.1. *If the BL updated expectation* $\hat{E}[Z_k|\mathbf{Y}]$ *for the unconstrained problem,* (4.3.1)*, is between* $[0,1]$*, then* $\tilde{\mathbf{h}}$*, the solution to the constrained problem* (4.3.8)*, is equal to* $\hat{\mathbf{h}}$*, the solution of the unconstrained problem. Otherwise one of the constraints to problem* (4.3.8) *is tight: if* $\hat{E}[Z_k|\mathbf{Y}] > 1$*, then the solution to the constrained problem satisfies* $\tilde{h}_0^k + \sum_{i=1}^n \tilde{h}_i^k y_i = 1$*; whilst if* $\hat{E}[Z_k|\mathbf{Y}] < 0$*, then* $\tilde{h}_0^k + \sum_{i=1}^n \tilde{h}_i^k y_i = 0$*.*

The proof for Lemma 4.3.1 is given in Appendix A.1.1. This allows us to motivate the following method for solving the constrained BL optimisation. Solving the unconstrained BL update, through (4.3.6) and (4.3.7), allows us to identify which constraint, if any, in (4.3.8) is tight. Once the tight constraint is identified, we can derive an analytical solution to the corresponding equality constrained problem. The benefit in computational time comes from the fact that solving (4.3.8) is reduced to several matrix multiplications, as opposed to using repeated gradient descent type methods required for general convex optimisation. More specifically, the algorithm to solve (4.3.8) for each $k = 1, \ldots, m$, is as follows.

1. Find the unconstrained BL updated expectation using (4.3.6).
2. If $\widehat{E}[Z_k \mid \mathbf{Y}]$ is between $[0,1]$, the solution to problem (4.3.8) is the same as that of the unconstrained BL update.
3. Otherwise compute the optimal solution of:

$$\begin{aligned} \underset{\mathbf{h}^k}{\text{minimise}} \quad & E\left[\left(Z_k - h_0^k - \sum_{i=1}^n h_i^k Y_i\right)^2\right], \\ \text{subject to} \quad & h_0^k + \sum_{i=1}^n h_i^k y_i = c, \end{aligned} \tag{4.3.10}$$

   with $c = 0$ if $\widehat{E}[Z_k \mid \mathbf{Y}] < 0$ and $c = 1$ if $\widehat{E}[Z_k \mid \mathbf{Y}] > 1$.

To solve problem (4.3.10) we can make use of the following lemma.

Lemma 4.3.2. *The analytical solution to the optimisation problem* (4.3.10) *is:*

$$(\tilde{\mathbf{h}}^k_{1:n})^T = \left(Cov(Z_k, \mathbf{Y}) + (c - E[Z_k])(\mathbf{y} - E[\mathbf{Y}])^T\right)\left(Var(\mathbf{Y}) + (\mathbf{y} - E[\mathbf{Y}])(\mathbf{y} - E[\mathbf{Y}])^T\right)^{-1},$$

and

$$\tilde{h}^k_0 = c - (\tilde{\mathbf{h}}^k_{1:n})^T\mathbf{y}.$$

The proof of Lemma 4.3.2 can be found in the Appendix A.1.1. The updated expectation and covariance can then be calculated from equations (4.3.2) and (4.3.3).

The computational cost of both the constrained BL and unconstrained BL, for a set of n observations and a set of m latent variables, is $O(m^2n + n^2m)$. This comes from the cost of solving the system of linear equations for $(h^k_{1:n})^T$ which takes $O(n^2)$ for each of the $k = 1, \ldots, m$ latent variables giving a cost of $O(mn^2)$ and calculating the updated covariance at a cost of $O(m^2n)$.

4.4 Bayes Linear for Network-Based Searches

The computational bottleneck of exact inference for the process is updating the beliefs on the participants' relevance values, $\mathbf{Z}$. We apply constrained BL updates to approximate the posterior mean and covariance of these latent variables. The constrained BL method for network-based search is given in Algorithm 6.

Algorithm 6 Bayes Linear Network-Based Search

1. Calculate approximations to quantities required for BL updates given the current set of observations. See Section 4.4.1.
2. Find $\tilde{E}[\mathbf{Z} \mid \mathbf{Y}]$ and $\tilde{\text{Var}}(\mathbf{Z} \mid \mathbf{Y})$ using constrained BL updates.
3. Calculate $\tilde{E}[\mathbf{P} \mid \mathbf{Y}]$ and $\tilde{\text{Var}}(\mathbf{P} \mid \mathbf{Y})$. See Section 4.4.2.
4. Decide which item to observe next, using $\tilde{E}[\mathbf{P} \mid \mathbf{Y}]$ and $\tilde{\text{Var}}(\mathbf{P} \mid \mathbf{Y})$.

4.4.1 Approximating BL Prior Values from $E[\mathbf{Z}]$, $\mathrm{Var}(\mathbf{Z})$ and $\mathbf{P}|\mathbf{Z}$

We assume the prior expectation and variance of the participants' relevance values are given, along with a prior conditional beta distribution for $\mathbf{P} \mid \mathbf{Z}$. Based on Section 4.3, performing BL updates also requires $E[\mathbf{Y}]$, $E[\mathbf{Y}\mathbf{Y}^T]$, $E[\mathbf{Z}\mathbf{Y}]$ and $E\left[\mathbf{Z}\mathbf{Z}^T\right]$. The value of $E\left[\mathbf{Z}\mathbf{Z}^T\right]$ can be calculated directly from the priors given to the user; the remaining quantities must be approximated, see Section 4.4.2. Furthermore, once the updated expectation and covariance of the $\mathbf{Z} \mid \mathbf{Y}$ are found, this method is used to approximate the updates for $\mathbf{P} \mid \mathbf{Y}$.

The observable quantities, $\mathbf{Y}$, are the number of relevant observations on each edge. The prior mean and covariance depends on the number of observations on each edge and the prior mean and covariance of the probabilities for each edge.

Lemma 4.4.1. *The expectations $E[\mathbf{Y}]$, $E[\mathbf{Y}\mathbf{Y}^T]$ and $E[\mathbf{Z}\mathbf{Y}]$ can be calculated analytically from $E[\mathbf{Z}]$, $Var(\mathbf{Z})$ and the prior conditional distribution for $\mathbf{P} \mid \mathbf{Z}$. The analytical solution for $E[\mathbf{Y}]$ and $E[\mathbf{Z}\mathbf{Y}]$ can be calculated from:*

$$E[Y_{uv}] = n_{uv} E[P_{uv}], \tag{4.4.1}$$

and:

$$E[Z_k Y_{uv}] = n_{uv} E[Z_k P_{uv}]. \tag{4.4.2}$$

where n_{uv} is the number of observations on edge (u, v) to date. For $E[\mathbf{Y}\mathbf{Y}]$, the diagonal entries are:

$$E\left[Y_{uv}^2\right] = n_{uv}(n_{uv} - 1) E\left[Var(P_{uv} \mid Z_u, Z_v) + E[P_{uv} \mid Z_u, Z_v]^2\right] + n_{uv}^2 E[P_{uv}], \tag{4.4.3}$$

whilst the off diagonal entries are given by:

$$E[Y_{uv} Y_{ij}] = n_{uv} n_{ij} E[P_{uv} P_{ij}]. \tag{4.4.4}$$

The proof of Lemma 4.4.1 is given in Appendix A.1.1.

4.4.2 Approximating Joint Distributions of Z

Some of the solutions in Lemma 4.4.1 require the joint distribution over several participants' relevance values and probabilities. However, we only have a prior mean and covariance of $\mathbf{Z}$ and the joint distribution for more than two binary Z values is not uniquely defined by their expectation and covariance. For a set of latent variables, we can approximate a possible joint distribution from the expectation and covariance matrix using BL updates. Given an ordered set of random variables $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_k)$, we construct a joint probability mass function as:

$$\tilde{p}(\mathbf{z}) = \tilde{p}(z_1)\tilde{p}(z_2 \mid z_1) \ldots \tilde{p}(z_k \mid z_1, \ldots, z_{k-1}). \tag{4.4.5}$$

For binary random variables the probability of observing a 1 is equal to the mean of the random variable. We approximate the conditional means using those obtained from the constrained BL updates, and thus obtain:

$$\tilde{p}(z_j|z_1, \ldots, z_{j-1}) = \tilde{E}[Z_j|z_1, \ldots, z_{j-1}]^{z_j}(1 - \tilde{E}[Z_j|z_1, \ldots, z_{j-1}])^{1-z_j} \tag{4.4.6}$$

and $\tilde{E}[Z_j|z_1, \ldots, z_{j-1}]$ is the updated expectation found using the constrained BL approximation. Chapter 5 looks in more detail at the method for calculating the joint probability distribution over a set of binary random variables. Alternatively, the maximum entropy distribution, described in Appendix A.1.2 can be used to approximate the joint distribution.

From this approximated distribution and the conditional beta distributions, we can calculate the required expectations. We describe here the calculations for $E[P_{uv}P_{ij}]$, with nodes u, v,

i and j being distinct:

$$\begin{aligned} E[P_{uv}P_{ij}] &= E\left[E\left[P_{uv}P_{ij}|Z_u, Z_v, Z_i, Z_j\right]\right] \\ &= \sum_{z_u,z_v,z_i,z_j\in\{0,1\}} \tilde{p}(z_u, z_v, z_i, z_j)E[P_{uv}|z_u, z_v]E[P_{ij}|z_i, z_j], \end{aligned}$$

as $E\left[P_{uv}P_{ij}|Z_u, Z_v, Z_i, Z_j\right] = E[P_{uv}|z_u, z_v]E[P_{ij}|z_i, z_j]$ by conditional independence and the joint probability $\tilde{p}(Z_u, Z_v, Z_i, Z_j)$ is calculated using 4.4.5 with the prior mean and covariance of the four random variables.

Similar calculations are used for $E\left[Z_k P_{uv}\right]$, $E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v)\right]$ and $E\left[E\left[P_{uv} \mid Z_u, Z_v\right]^2\right]$, and these are given in Appendix A.1.3. These quantities need to be calculated for steps (1) of Algorithm 6. The calculation for step (1) uses the prior distributions, and thus can be carried out just once regardless of the number of items processed.

For step (3) we need to use the posterior BL mean and covariance for $\mathbf{Z}$, $\tilde{E}(\mathbf{Z} \mid \mathbf{Y})$ and $\tilde{\mathrm{Cov}}(\mathbf{Z} \mid \mathbf{Y})$, to obtain the BL estimate of the posterior mean and variance for $\mathbf{P}$. For an ordered set of random variables $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_k)$, given observations $\mathbf{Y} = \mathbf{y}$, we construct an approximate joint posterior probability mass function as:

$$\tilde{p}(\mathbf{z} \mid \mathbf{y}) = \tilde{p}(z_1 \mid \mathbf{y})\tilde{p}(z_2 \mid z_1, \mathbf{y}) \ldots \tilde{p}(z_k \mid z_1, \ldots, z_{k-1}, \mathbf{y}). \tag{4.4.7}$$

The conditional probability $\tilde{p}(z_j \mid z_1, \ldots, z_{j-1}, \mathbf{y})$ is obtained from:

$$\tilde{p}(z_j|z_1, \ldots, z_{j-1}, \mathbf{y}) = \tilde{E}[Z_j|z_1, \ldots, z_{j-1}, \mathbf{y}]^{z_j}(1 - \tilde{E}[Z_j|z_1, \ldots, z_{j-1}, \mathbf{y}])^{1-z_j} \tag{4.4.8}$$

where $\tilde{E}[Z_j|z_1, \ldots, z_{j-1}, \mathbf{y}]$ is calculated using constrained BL update equations, using the constrained BL expectation and covariance of $\mathbf{Z} \mid \mathbf{y}$.

From this approximated distribution and the conditional posterior beta distributions, we can approximate the full posterior joint distribution given in (4.2.4) and calculate the required posterior estimations. For example, to calculate the posterior expectation, $E[P_{uv}P_{ij} \mid \mathbf{y}]$, with nodes u, v, i and j being distinct, requires calculating the posterior joint distribution

$\tilde{p}(Z_u, Z_v, Z_i, Z_j \mid \mathbf{y})$ using (4.4.7) and then:

$$\begin{aligned} E[P_{uv}P_{ij} \mid \mathbf{y}] &= E\left[E\left[P_{uv}P_{ij}|Z_u, Z_v, Z_i, Z_j, \mathbf{y}\right]\right] \\ &= \sum_{z_u,z_v,z_i,z_j \in \{0,1\}} \tilde{p}(z_u, z_v, z_i, z_j \mid \mathbf{y}) E[P_{uv}|z_u, z_v, \mathbf{y_{uv}}] E[P_{ij}|z_i, z_j, \mathbf{y_{ij}}]. \end{aligned}$$

Similar calculations are used for $E\left[Z_k P_{uv} \mid \mathbf{y}\right]$, $E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v, \mathbf{y})\right]$ and $E\left[E\left[P_{uv} \mid Z_u, Z_v, \mathbf{y}\right]^2\right]$, and these are given in Appendix A.1.4. The calculations in step (3) would need to be repeated before choosing each item.

4.5 Results

In this section, we analyse the BL model for the network-based search problem. Firstly, for small networks we consider the errors induced using the BL model as an approximation to the binary MRF model, and observe how this affects the performance of the decision problem. We empirically show these errors are small and that there is little difference in the performance for a range of sequential decision policies if we use the BL model rather than the MRF model.

4.5.1 Bayes Linear as an Approximation to the MRF Model

To evaluate the accuracy of the BL models as an approximation to the binary MRF model, we assume the true underlying model for the networks is a binary MRF model. We consider a model where the joint probability of $\mathbf{Z}$ is proportional to the product of a set of factors, with a factor associated with each edge in the network. We use factors of the form in Table 4.5.1a. For $\lambda_1 > 0$ and $\lambda_2 > 0$ the network exhibits the property of homophily: the binary random variables on nodes that are connected by an edge are more likely to be of the same value. The larger λ_1 or λ_2 are, the more likely the random variables will both be 0 or both be 1.

Z_i	Z_j	$\phi(Z_i, Z_j)$
0	0	$1 + \lambda_1$
0	1	1
1	0	1
1	1	$1 + \lambda_2$

(a) Prior Clique Factor

Z_i	Z_j	$a(Z_i, Z_j)$	$b(Z_i, Z_j)$
0	0	1	9
0	1	1	4
1	0	1	4
1	1	1	1

(b) Prior Conditional 1

Z_i	Z_j	$a(Z_i, Z_j)$	$b(Z_i, Z_j)$
0	0	1	9
0	1	1	4
1	0	1	4
1	1	9	1

(c) Prior Conditional 2

Table 4.5.1: The prior distributions used to define the binary MRF model. Table 4.5.1a gives form of the prior clique factor used to define the prior MRF model for the participants. Tables 4.5.1b and 4.5.1c give the two parameters of the conditional beta distributions used to define $\mathbf{P}|\mathbf{Z}$.

We model the conditional probability of observing a relevant item on an edge as a beta distribution which is dependent on the involved nodes relevance values. Two conditional prior distributions are considered. We call these prior conditional 1 (Table 4.5.1b) and prior conditional 2 (Table 4.5.1c). Prior conditional 1 is more skewed to the belief that we are less likely to observe relevant items between participants even when both participants are considered relevant to the query.

The prior mean and covariance required for the BL model are set equal to those calculated from the binary MRF model. The BL updated expectations and variances are calculated for a sequence of observations. These updated expectations and variances are compared to the corresponding values in the binary MRF model updated using exact inference.

Simple Line Network

Firstly, we consider a simple line network with three nodes and two edges. The edges connect Z_1 with Z_2 and Z_2 with Z_3. The binary MRF model is defined using the prior clique factor in Table 4.5.1a with $[\lambda_1, \lambda_2] = [0.5, 0.5]$ and using prior conditional 1 (Table 4.5.1b). Figures 4.5.1 and 4.5.2 shows the updated expectations and variances for both

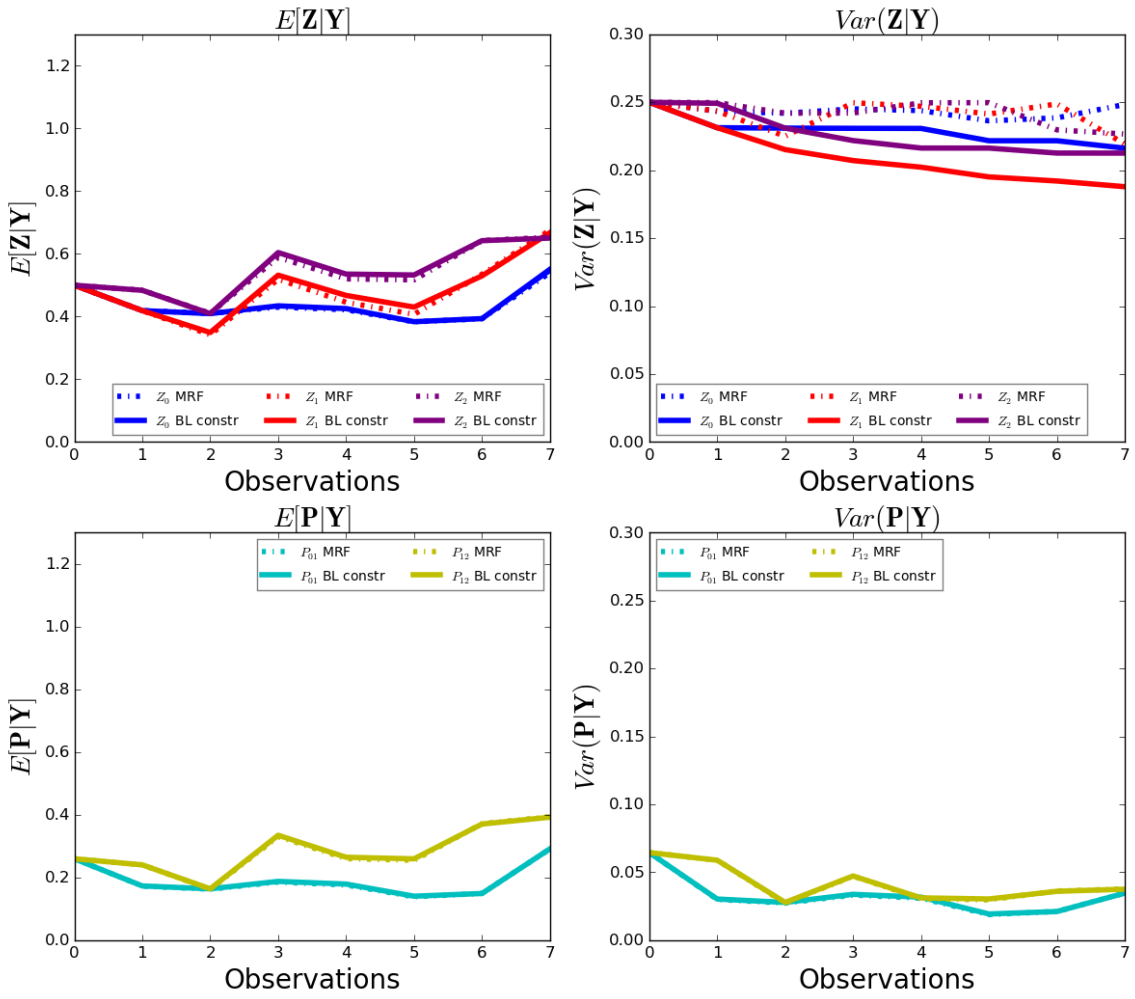


Figure 4.5.1: The updated expectations and variances of the participants' relevance values, $\mathbf{Z}$, and probabilities, $\mathbf{P}$, using the MRF, unconstrained BL, and constrained BL models with set of observations $\mathbf{Y} = (Y_{01}^1 = 0, Y_{12}^1 = 0, Y_{12}^2 = 1, Y_{12}^3 = 0, Y_{01}^2 = 0, Y_{12}^4 = 1, Y_{01}^3 = 1)$. The prior clique factor is given in Table 4.5.1a with $[\lambda_1, \lambda_2] = [0.5, 0.5]$, and prior conditional probability distribution in Table 4.5.1b. The prior values for BL models are calculated directly from the binary MRF prior model.

$\mathbf{Z} \mid \mathbf{Y}$ and $\mathbf{P} \mid \mathbf{Y}$ using the constrained BL model, unconstrained BL model and the binary MRF model for two sets of observations. The unconstrained BL model and the constrained BL model give the same updated values for the set of observations used in Figure 4.5.1. These values remain close to the values from the binary MRF model apart from $\text{Var}(\mathbf{Z}|\mathbf{Y})$. However, for binary random variables, the mean determines the variance of the variable, so these values are somewhat redundant.

Figure 4.5.2 shows a set of observations for which the constrained updates ensure the up-

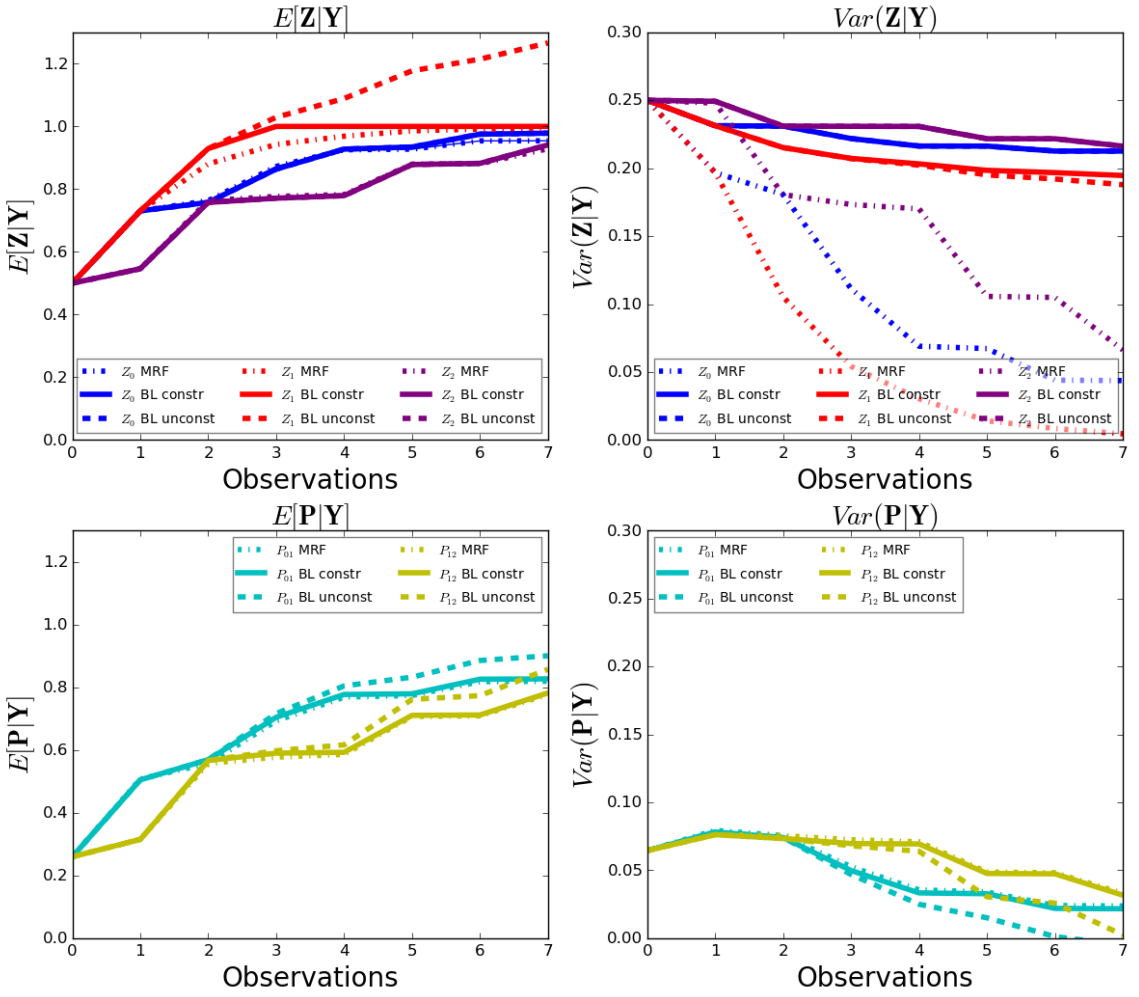
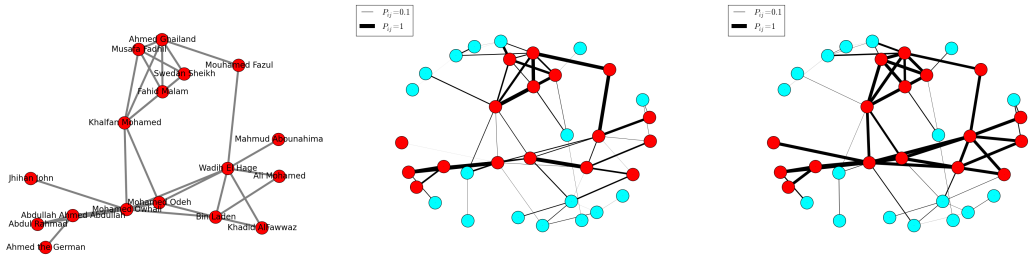


Figure 4.5.2: he updated expectations and variances of the participants' relevance values, $\mathbf{Z}$, and probabilities, $\mathbf{P}$, using the MRF, unconstrained BL, and constrained BL models with set of observations $\mathbf{Y} = (Y_{01}^1 = 1, Y_{12}^1 = 1, Y_{12}^2 = 1, Y_{12}^3 = 1, Y_{01}^2 = 1, Y_{12}^4 = 1, Y_{01}^3 = 1)$. The prior clique factor is given in Table 4.5.1a with $[\lambda_1, \lambda_2] = [0.5, 0.5]$, and prior conditional probability distribution in Table 4.5.1b. The prior values for BL models are calculated directly from the binary MRF prior model.

dated expectation of $\mathbf{Z}|\mathbf{Y}$ remains in the correct range. As well as giving a more mathematically elegant solution, the benefit of the constrained BL model can be seen in the updated expectation and variance of $\mathbf{P}|\mathbf{Y}$, where the constrained BL updates are closer to the updates in the binary MRF than when the unconstrained updates are used.



(a) Tanzania network

(b) Network: Prior Conditional 1

(c) Network: Prior Conditional 2

Figure 4.5.3: The Tanzania network used to test the accuracy of constrained BL updates. Figure 4.5.3a shows a possible social network behind the terrorists responsible for the 1998 bombing of the US embassy in Tanzania. Figure 4.5.3b and 4.5.3c show the full networks used to test the BL model. The thickness of the edge represents the probability of observing a relevant item on that edge, sampled from prior conditional 1 and prior conditional 2 for Figure 4.5.3b and 4.5.3c respectively. The terrorists (irrelevant participants) are represented by red (blue) nodes and have true relevance 1 (0).

Tanzania Network

We return to the Tanzania network presented in Section 4.1. The network, shown again in Figure 4.5.3a, is a possible terrorist network associated with the bombing of the US embassy in Tanzania in 1998 (Nevo, 2011). The network consists of 17 terrorists involved in the plot and is generated based on information from the Carnegie Mellon Computational Analysis of Social and Organisational Systems Laboratory (2004). In addition to the 17 terrorists, we also consider 17 irrelevant participants. Edges are added randomly between an irrelevant participant's node and other nodes in the networks. The true probability of observing a relevant intelligence item for each edge is then sampled given the involved participants' true relevancies. We simulate two realisations of the edge probabilities, one using prior conditional 1 and one using prior conditional 2. The networks and probabilities are shown in Figures 4.5.3b and 4.5.3c.

For 250 sets of 300 observations, the updated expectations and variances are sequentially calculated in both the binary MRF model and constrained BL model (henceforth simply called the BL model). The observations are on randomly selected edges and the relevance of the item is sampled from the true probability of observing a relevant item on that edge. We look at the distribution of differences in the estimated posterior means and covariances

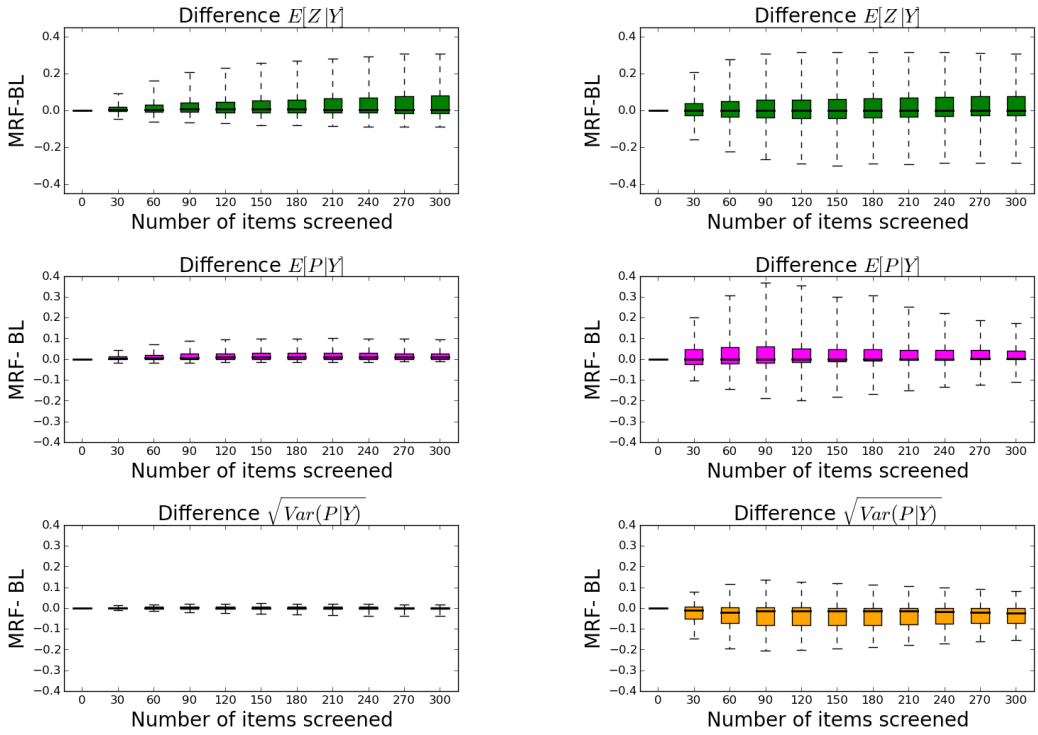
between the BL and MRF models. These are shown in Figure 4.5.4 after a given number of observations on all nodes or edges in the network over all 250 sets of observations.

The binary MRF model is defined using the clique factor in Table 4.5.1a with $[\lambda_1, \lambda_2] = [0.5, 0.5]$ and using both prior conditional 1 and prior conditional 2. For both prior conditional distributions, the difference between the BL model and the MRF model is small for the expectation of $\mathbf{Z}|\mathbf{Y}$, with the majority having an absolute difference of less that 0.1 after 300 observations. The symmetry of the prior conditional distributions are reflected in the shape of the distribution of differences for $\mathbf{Z}|\mathbf{Y}$. Prior conditional 2, which is more symmetric also has more symmetric differences, see Figure 4.5.4b. Using prior conditional 1, the BL model is more likely to underestimate the expectation than overestimate, see Figure 4.5.4a.

The accuracy of BL updates for $\mathbf{P} \mid \mathbf{Y}$ is affected more by the prior conditional probability. There are only very small differences in the binary MRF and BL model updates of $\mathbf{P} \mid \mathbf{Y}$, when using prior conditional 1, compared with prior conditional 2. Prior conditional 2 is more dependent on the involved participants so a small error in the constrained BL expectation of $\mathbf{Z} \mid \mathbf{Y}$ will have a larger effect on the constrained BL updates of $\mathbf{P} \mid \mathbf{Y}$. Hence the accuracy of the BL approximation is at least partially dependent on the model choice for the conditional probability distribution.

Sequential Decision Problem

Our real interest is the effect of using the BL model, rather than the MRF model, on the accuracy of decisions about which items to screen. To evaluate this for the Tanzania networks used in Section 4.5.1, we simulate items to associate with each edge. The number of items on each edge is drawn, independently, from a Poisson distribution with mean 30. Each item is relevant, independently of all other items, with a probability equal to the probability of that edge. We compare three heuristic policies for choosing with items to screen: greedy, ϵ-greedy and Bayes-UCB (Kaufmann et al., 2012a). The greedy policy is a pure exploitation method, that will choose an item at random from the edge with the highest



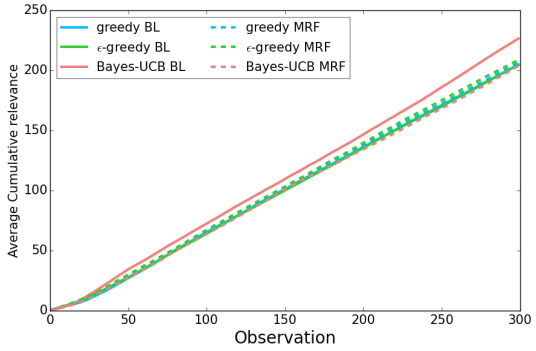
(a) Prior conditional 1: Differences

(b) Prior conditional 2: Differences

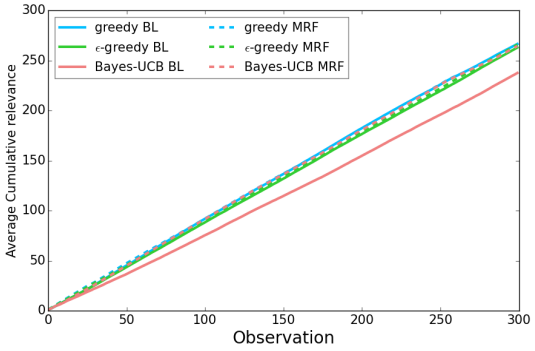
Figure 4.5.4: The box plots show, for a given number of observations along the x-axis, the distribution of differences between the expectations and variances in the binary MRF model and using the BL model for inference in the Tanzania network. Figure 4.5.4a show the results for prior conditional 1, corresponding to the network in Figure 4.5.3b, and Figure 4.5.4b show the results for prior conditional 2, corresponding to the network in Figure 4.5.3c. Each box plots show the median difference, and interquartile range. The whiskers show the 95% quantiles interval for the differences.

posterior mean probability. The ϵ-greedy policy incorporates some additional exploration. At each iteration it uses the greedy policy with probability $1 - \epsilon$, and with probability ϵ selects an item from a randomly chosen edge.

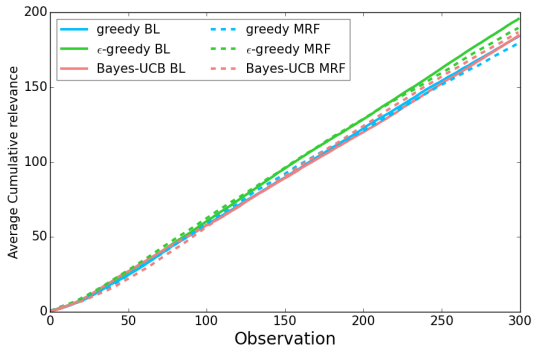
The greedy and ϵ-greedy heuristic policies do not consider uncertainty in the probability of observing a relevant item on each edge. By comparison the Bayes-UCB policy uses information from the posterior variances. This policy chooses items on the edge which has the highest value of some user-chosen quantile of the posterior distribution of the probabilities associated with each edge. The Bayes-UCB policy has strong similarities to UCB and its variants (Auer et al., 2002; Garivier and Cappé, 2011). We implement this method by approximating the posterior distribution of each probability by a Gaussian distribution



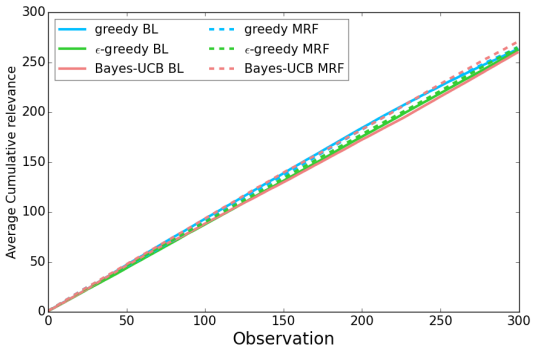
(a) Network: Prior Conditional 1, Prior: Prior Conditional 1



(b) Network: Prior Conditional 2, Prior: Prior Conditional 2



(c) Network: Prior Conditional 1, Prior: Prior Conditional 2



(d) Network: Prior Conditional 2, Prior: Prior Conditional 1

Figure 4.5.5: The average cumulative number of relevant items found with different heuristic methods, using both the BL model and binary MRF model. The dotted lines are the cumulative number of relevant items observed using binary MRF model and the solid line is when the BL model is used. The decision policies are run with both the correct prior model for the probabilities and the wrong model. There is very little difference between the performance of the heuristic methods for the BL model and binary MRF model and the choice of prior model does not seem to have a significant effect on the performance of the decision policy.

with the BL estimate of the posterior mean and variance. At time t the upper confidence bound of $1 - \alpha_t$ with $\alpha_t = 1/t$ is chosen because of its good performance in simulations by Kaufmann et al. (2012a), for both binary rewards and Gaussian rewards. When α_t is of the order $1/t$, Bayes-UCB can be shown to achieve asymptotic optimality for binary rewards.

Each heuristic method for the search process is run 50 times on the networks shown in Figure 4.5.3b and 4.5.3c. Figure 4.5.5a shows the average cumulative number of relevant items observed over the 50 repetitions using prior conditional 1, for the different heuristic methods using both the binary MRF model and BL model. In general, all heuristic algorithms give

very similar results. The network structure of the model means that even through the greedy heuristic method we learn about the probability of observing a relevant item on other edges in a network. For small networks, this means we will quickly find the areas of the network with relevant observations, no matter which heuristic algorithm is used.

For the greedy and ϵ-greedy heuristic method, using the BL model as opposed to the binary MRF model results in roughly the same average number of relevant items observed. Whilst the differences between expectations in the binary MRF model and BL model for prior conditional 2 were larger for $\mathbf{P} \mid \mathbf{Y}$, these errors have little effect on the performance of the greedy and ϵ-greedy heuristic methods, shown in Figure 4.5.5b. This suggests the BL updates may capture enough of the updating process to perform well when using these two search policies.

The results for the Bayes-UCB algorithm are more varied for the two models. Figure 4.5.4b shows that for a random set of observations the BL model tends to overestimate the posterior standard deviation. This suggests the BL model will give a higher upper confidence bound compared to the MRF model which, in this case, results in slightly fewer relevant items being found, see Figure 4.5.5b. However, for prior conditional 1 any errors in posterior values from using the BL model result in the Bayes-UCB model giving superior performance when compared to using either the MRF model or the BL model together with other search policies. Thus we see that the errors induced in the BL model are not always detrimental to the performance of the decision algorithm.

To analyse the robustness of the decision policies to the choice of prior model, we analyse the networks with the wrong model for the prior conditional probabilities on the edges, see Figures 4.5.5c and 4.5.5d. We see that the choice of model for the prior conditional probabilities on the edges does not have a significant effect of the performance of the decision policies. The only noticeable change is that the performance of the Bayes-UCB decision policy with the BL model is now more in line with the other decision policies.

4.6 Robustness to the Amount of Dependence

For larger networks we cannot compare results with the binary MRF model because it is computationally intractable. Thus to benchmark performance of our BL model we compare to a simple model which assumes independence between each edge. We show that even when there is little correlation in the network, the BL model is not detrimental to the performance of the decision algorithm. When the networks are correlated, the BL model results in a higher number of relevant items being observed.

To benchmark the performance of the BL model approach we will compare to a simpler approach that treats each edge in the network independently. Such an approach is able to scale computationally to very large networks, and thus is a natural competitor as we consider analysing larger networks for which the MRF model is impracticable. Our aim here is to see whether modelling the dependence of the nodes in the network does lead to more accurate decisions when there is dependence, and whether there is a loss of accuracy in using the BL model when the relevance values of the nodes are actually independent of one another.

Our alternative model for the data, which we henceforth call the *independence model*, treats all edges independently. It assumes a prior distribution for the probability associated with an edge that is a mixture of beta distributions. The choice of mixture components is fixed to be the distributions that are assumed in the BL model for each of the possible realisation of nodes values. The model is then completed by specifying the weights for each of these components. It is straightforward to analytically update this prior distribution for the probability on an edge each time we screen an item from that edge. Under this model, items screened on one edge will only impact the posterior distribution for the probability of that edge – they have no effect on the posterior distributions for probability on any other edges.

The method used to simulate the nodes' relevance values is given in Section 4.6.1. One issue we have to address when implementing the BL model for this case is how we choose an

appropriate prior mean and covariance. The approach we take is described in Section 4.6.2 for the BL model. In Section 4.6.3 we describe how we choose the prior mixture weights for the independence model so as to closely match the BL model. Finally, a comparison of the updated method's performance in the decision process for the simulated networks is given in Section 4.6.4.

4.6.1 Simulating Correlated Node Relevancies

The method we use to simulate correlated random variables on the network is to "infect" a node's neighbours with the same relevance value with some probability ρ.

1. Pick an initial node, Z_i.
2. With probability 0.5, let $Z_i = 1$; otherwise let $Z_i = 0$.
3. For each node j which is a neighbour of node i and has not yet been assigned a relevance value, let

$$Z_j = \begin{cases} z_i & \text{with probability } \rho, \\ 1 - z_i & \text{with probability } 1 - \rho. \end{cases} \tag{4.6.1}$$

4. Randomly select an infected node with neighbours to infect, and go to step 3.

A value of $\rho = 0.5$ will give a random allocation of node relevancies. Increasing ρ will lead to increased correlation within the network.

For the purpose of testing the effect of correlation in the network on the different models' performance, given the node relevancies, we set the probability of observing a relevant item on an edge to:

$$P_{i,j} = \begin{cases} 0.0 & \text{if } z_i + z_j = 0 \\ 0.2 & \text{if } z_i + z_j = 1 \\ 0.9 & \text{if } z_i + z_j = 2 \end{cases} . \tag{4.6.2}$$

4.6.2 Bayes Linear Prior Expectation and Covariance

The BL model requires prior specification of the mean and covariance of $\mathbf{Z}$. In Section 4.5.1 these are calculated directly from the binary MRF model prior. When this model is not assumed to be the true underlying model, or on larger networks where calculating directly from the binary MRF is computationally prohibitive, we need a way to choose the prior mean and covariance.

For simplicity we assume the prior mean is the same value, μ, for all nodes. To define the prior covariance matrix we consider the network structure (Loh et al., 2013), and relate this to the prior precision matrix. In particular, for all pairs of nodes, i and j for example, that are not connected by an edge, we impose the condition that the ijth entry of the precision matrix is 0. Using a factor, which controls the strength of correlation between random variables, $\delta \in (-1, 1)$, a simple specification of the covariance is then given by

$$\Sigma = B^T Q^{-1} B, \tag{4.6.3}$$

where:

$$Q_{ij} = \begin{cases} 1 & : i = j \\ \frac{-\delta}{max(n_i, n_j)} & : i \in ne(j) \\ 0 & : i \notin ne(j) \end{cases}, \tag{4.6.4}$$

$$B = \begin{pmatrix} \sqrt{(\mu(1-\mu))/Q_{1,1}^{-1}} & 0 & \dots & 0 \\ 0 & \sqrt{(\mu(1-\mu))/Q_{2,2}^{-1}} & \dots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \dots & 0 & \sqrt{(\mu(1-\mu))/Q_{m,m}^{-1}} \end{pmatrix}, \tag{4.6.5}$$

and n_i is the number of neighbouring nodes of node i. This method will ensure the diagonal entries of Q are positive and the matrix is diagonally dominant; this gives sufficient conditions for the matrix to be positive definite and hence ensures a positive definite co-

variance matrix. It also ensures that the prior variance is consistent with the prior mean: $\Sigma_{ii} = \mu(1-\mu)$, for all i.

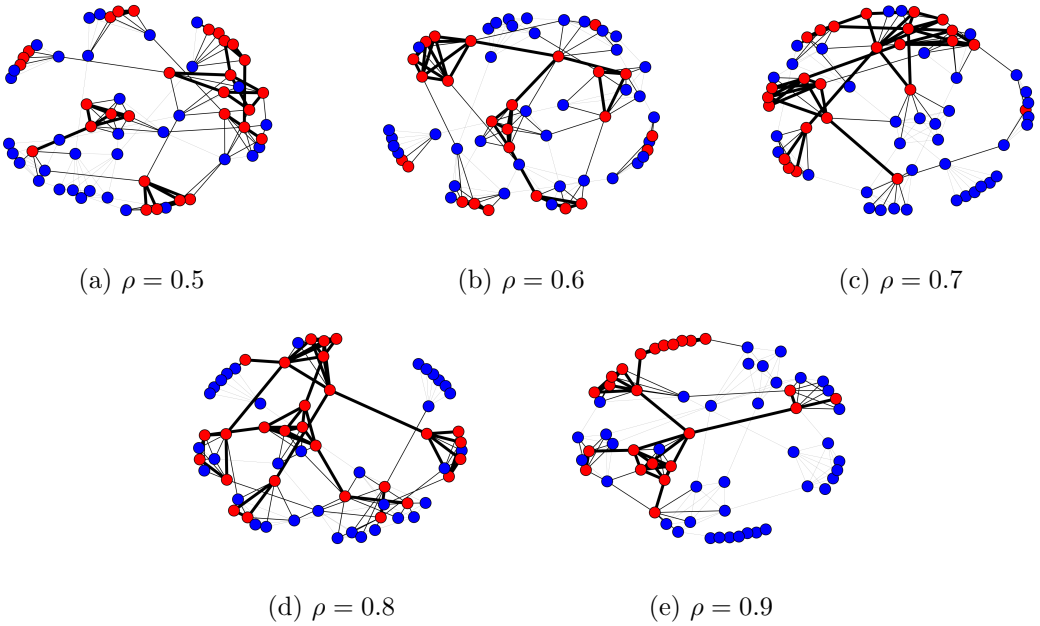
4.6.3 Varying the Prior for the Independence Model

We also need to specify a prior for the independence model to which we compare the BL model. For ease of comparison we try to match the priors of the two models, so they both have the same prior expectation. However, for any value of prior expectation $E[Z] = \mu$, there is a continuous range of possible prior distributions for the independence model. If we let P_{ij} denote the prior probability of $Z_1 = i$ and $Z_2 = j$, then we impose the restrictions of symmetry, $P_{01} = P_{10}$, and that the mean matches the BL model mean: $P_{10} + P_{11} = \mu$. This leaves one degree of freedom. In order to ensure this prior distribution is not influencing the performance of the independence model when compared to the BL model, we can vary this final degree of freedom and run the independence model with a range of prior distributions. We do this by varying the value of the 2nd moment $E[Z_i Z_j] = P_{11}$ between $0 \leq P_{11} \leq \mu$.

4.6.4 Results for Simulated Networks

We look at the performance of the network-based search method for a network simulated using the `relaxed_caveman_graph` function in NetworkX (Hagberg et al., 2008). The true relevance of nodes is defined using the method in Section 4.6.1 for a range of ρ values. For a value of ρ, the same network is used in all iterations, see Figure 4.6.1. We would expect the BL model to have superior performance for networks where there is larger correlation in the nodes' random variables, which corresponds to larger values of ρ.

For both the independence model and the BL model, we set the prior conditional probability distribution to that of Table 4.5.1c. The prior mean for the BL model is set to either 0.25 or 0.5 and the covariance is defined using the method in Section 4.6.2 with $\delta = 0.8$. The independence model is run for a range of prior distributions as described in Section 4.6.3, matching the mean of the BL model. Figure 4.6.2 shows the mean total number of relevant



(a) $\rho = 0.5$ (b) $\rho = 0.6$ (c) $\rho = 0.7$

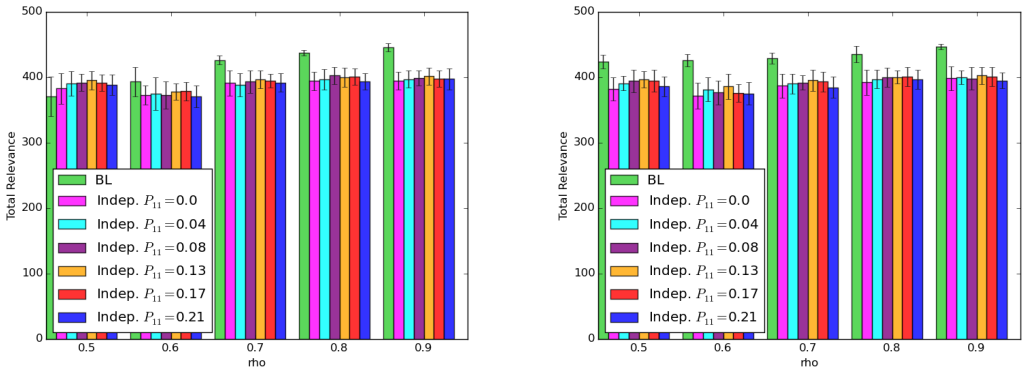
(d) $\rho = 0.8$ (e) $\rho = 0.9$

Figure 4.6.1: Relaxed caveman network and simulated relevance values of nodes, for different values of ρ using the method in Section 4.6.1. The red (blue) nodes represent relevant (irrelevant) participants. For higher values of ρ, the nodes' relevance values should be more clustered.

observations for the BL model and for the independence model, using the greedy decision policies over 50 repetitions with with 500 observations each. We see that for all cases where there is dependence in the latent variables associated with the nodes, that is $\rho \geq 0.6$, the BL model does noticeably better than the independence model. There is also some evidence that the improvement from using the BL model is greater for larger values of ρ. Furthermore, even when there is no correlation in the networks, corresponding to $\rho = 0.5$, use of the BL model tends not to be detrimental.

4.7 Enron Network Data

In this section, we look at how the different models perform when applied to communication networks simulated from the Enron Corpus. The Enron Corpus consists of hundreds of thousands of emails from roughly 150 senior employees at the Enron Corporation. However the total number of nodes in the full network is much larger as it includes people from outside



(a) greedy: $\mu = 0.25$ (b) greedy: $\mu = 0.5$

Figure 4.6.2: The mean and ± 1 standard error bars of the total number of relevant items found over 500 observations for the greedy policy run on networks simulated using the `relaxed_caveman_graph` function in NetworkX (Hagberg et al., 2008), shown in Figure 4.6.1. The independence models are run for a range of second moments with the mean matching the BL model mean.

Enron who received or sent emails in the Corpus. The full Enron network contains over 28,000 nodes. The dataset was made public during the US government's legal investigation of the Enron Corporation after its collapse in 2001. Our interest in this data is that it has similarities with the datasets of corporate legal cases, such as the *Oracle America, Inc. v. Google Inc.* case.

Following Ellis (2013) we classify an email as relevant if it contains the words "Money" or "Finance". We wish to consider networks of differing sizes. Using an approach from Ellis (2013), subsets of the full Enron network are obtained. First an initial number of nodes with high relevance are chosen to be in the network; Then, all edges and any nodes associated with those edges are added to the network with some probability. There then follows further iterations of selecting a node and adding new edges and their corresponding nodes with some probability. This is stopped when a pre-specified maximum number of nodes have been added. Finally, any nodes with degree one are trimmed with some probability to give a denser network. For these types of networks exact inference methods are likely to be slow.

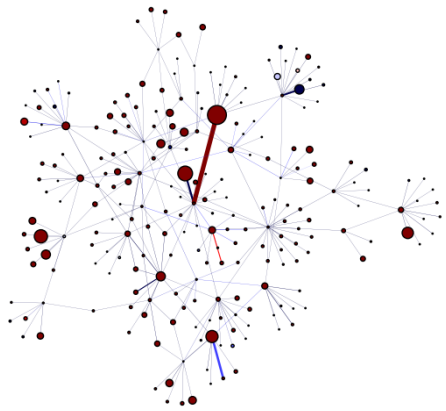
Using this approach we simulated data sets corresponding to networks of differing sizes.

For our smallest network, which only has 234 nodes, we can still run exact inference under the MRF model. For that network we compare the number of relevant items found using the BL model, the MRF model and the independence model; see Section 4.7.1. For larger networks, we only compare the number of relevant items found using the BL model and the independence model; see Section 4.7.2.

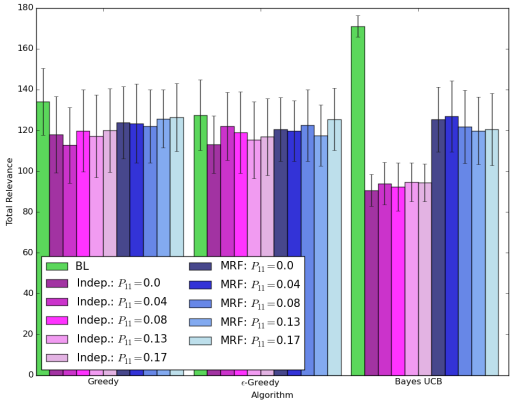
4.7.1 Small Enron Network

A small network was simulated using the method described in Section 4.7, where there is one initial node, edges connected to the node are selected with probability 0.04 and the maximum number of nodes is set to 300. Nodes with degree 1 were trimmed with probability 0.5. The network has 234 nodes and 275 edges and is shown in Figure 4.7.1a. There are a total of 4958 emails in the network, of which only 260 are relevant to the query. For the prior BL model we set $E[Z] = 0.25$ and use an approximate covariance with $\delta = 0.8$. Both the binary MRF model and the independence model are run with a range of priors for the $\mathbf{Z}$, which are found by matching the mean of the BL model and varying the second moment of the clique factor, see Section 4.6.3.

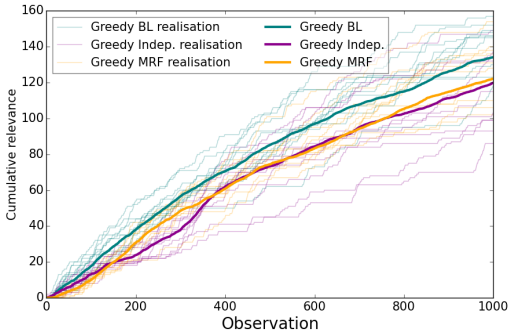
We run 30 repetitions with 1000 observations of each repetition. For each decision policy, the BL model finds more relevant items than the MRF model, which in turn finds more relevant items than the independence model; see Figure 4.7.1b. Using the Bayes-UCB policy with the BL model gives far superior performance to the other two models with this heuristic. The independence model does particularly badly with this policy as it does not consider the information learnt when screening observations on neighbouring edges. Figure 4.7.1c and 4.7.1d shows the mean number of relevant observations and the number of times a change of edge is made in the algorithm for the three models using the greedy policy. The BL model tends to remain on edges for longer than the independence model or the MRF model. For the Enron network, where there is no true underlying model, the less computationally intensive BL model gives better performance that the MRF model, suggesting it gives a better approximation to the true network behaviour. The MRF model may be making



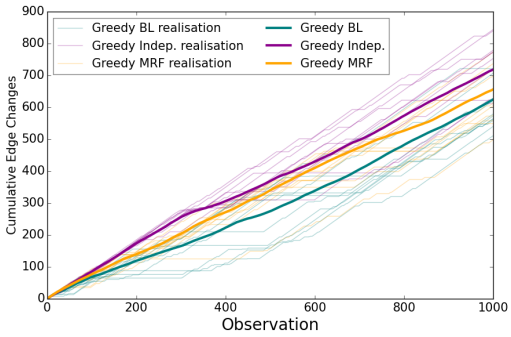
(a) Enron Network



(b) Bargraph of mean total relevance ±1 standard error



(c) Average Cumulative Relevance: greedy policy



(d) Number of Edge Changes: greedy policy

Figure 4.7.1: Results for the small Enron network, shown in Figure 4.7.1a. Figure 4.7.1b shows the mean and ±1 standard error bars of the total number of relevant items found over 1000 observations for different models and heuristic policies. The mean cumulative relevance over 1000 observations is shown in Figure 4.7.1c for the greedy policy with $P_{11} = 0.13$. Figure 4.7.1d shows the mean number of times the policy changes edges. The BL model performs better than both the MRF and independence model for each of the decision policies.

incorrect assumptions about the network behaviour leading to the decision policies moving off the relevant edges after then have been found and resulting in a higher number of edge changes.

4.7.2 Larger Enron Networks

For larger networks, the binary MRF model is computationally intractable. We compare the BL model and independence model on three networks shown in Figure 4.7.2. These have have 448 nodes and 630 edges; 669 nodes and 1133 edges; and 1641 nodes and 1957

edges respectively. We set the prior mean to $E[Z] = 0.25$ in the BL model with the prior covariance approximated, using the method in Section 4.6.2 with $\delta = 0.8$. The independence model is run with a range of priors by fixing $E[Z_i Z_j] = P_{11}$ over the range of $0 < P_{11} < \mu$ where $\mu = 0.25$. The range of P_{11} values is given by $P_{11} = [0.0, 0.04, 0.08, 0.13, 0.17, 0.21]$.

Results are shown in Figure 4.7.2, where 50 repetitions were run for each of the decision policies with 2000 observations of each. For the two smaller networks we get very similar results for the total number of relevant items screened when we use the independence model regardless of the value of P_{11}. The BL model gives superior performance to the independence model.

For the largest Enron network, the independence model's performance depends on the value of P_{11}. The model where $P_{11} = 0.0$ performs best; that is when we assume there is zero probability that both nodes involved are relevant to the query. Again, the BL model performs uniformly better than the independence model.

All three decision policies give similar results. However, as the number of nodes in the network increases, Bayes-UCB tends to do worse compared to the greedy methods. Bayes-UCB will spend longer exploring the network as the number of edges increases, which for the relatively short time horizon will not be advantageous. The network shown in Figure 4.7.2e has a cluster of relevant edges in one part of the network. If the greedy or epsilon greedy policy finds an area with a high concentration of relevant edges, it will stick there, whereas Bayes-UCB will likely move away again.

To help understand why the BL model is giving better performance than the independence model, we look more closely at the results for the greedy heuristic for the network shown in Figure 4.7.2c, and focus on the independence model with $P_{11} = 0.13$. Figure 4.7.3a shows the mean cumulative number of relevant items that are screened, averaged over 50 runs, as well as 10 realisations of the cumulative number of relevant items screened. The BL model generally finds the relevant items faster than the independence model. The cumulative number of times the algorithm changes edges is shown in Figure 4.7.3b for the two models. The BL model tends to change edges less often, remaining on the same edge for longer

periods of time. Towards the end of the runs, the independence model starts remaining on edges more often. This corresponds to an increase in the number of relevant items found: after a while the independence model finds edges with high relevance value.

Figure 4.7.3c and Figure 4.7.3d show the distribution of times that each algorithm selects an edge with a true probability greater than the 95th and 90th quantile respectively. For this network, those values correspond to probabilities of 0.5 and 0.124. Although the algorithm with the independence model sometimes selects more edges with probabilities greater than 0.5 than the BL model, performance is more varied. Furthermore, when we look at the edges with probability above the 90th quantile, the BL model selects a far larger number of these edges, generally picking edges above the 90th quantile over half the time, compared to the independence model which selects these edges less than half the time. The sharp jumps up in Figure 4.7.3e suggest that when the BL model finds a good edge, it will stick on the good edge for a while, before moving on and finding another edge. There are far more smaller jumps in the independence model: even when it finds a good edge it does not necessarily remain on it. The steep gradient of Figure 4.7.3f for the BL model suggest that when the algorithm moves off a very good edge it is likely to move to another one which is still good (above the 90th quantile).

Changing the Utility Function

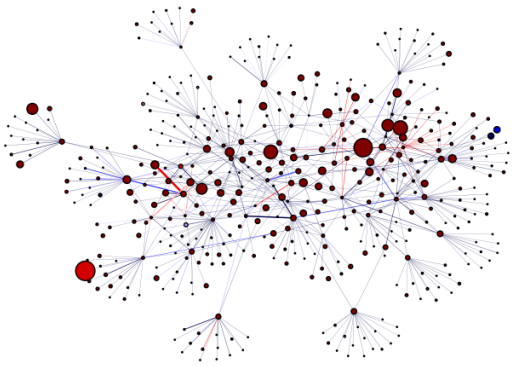
We consider a different utility function for the decision problem to help understand the behaviour of the model. If we are only interested in finding edges which have at least one relevant item, the decision algorithm can be changed so that when a relevant item is found, no more items are observed on that edge. This will give a better indication of how well the BL model and the independence model find a relevant edge, after finding a relevant item on another edge. The independence model and BL model are run for this utility function, with a horizon of 500 items on the network show in Figure 4.7.2c, with the same priors as in Section 4.7.2. The number of edges on which we observe a relevant item is shown in Figure 4.7.4a. The BL model observes far more relevant items than the independence model

using the greedy policy, suggesting that the BL model does a better job of finding the edges which contain relevant items. Furthermore, Figure 4.7.4b and 4.7.4c show number of times an item is observed on a "good" edge where "good" is taken as the top 95th quantile and top 90th quantile respectively. The BL model better focuses on the edges where observing relevant items is more likely. The decision algorithm pick far more good edges for the BL model than the independence model.

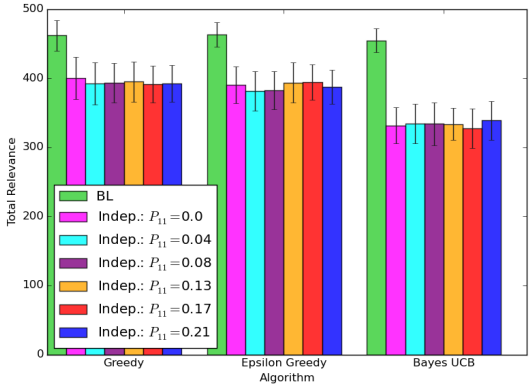
4.8 Discussion

We have considered the problem of searching a network of communications to find those items which are relevant to a query. We show that the BL methods provide a natural approach to modelling such data. We introduce a new optimisation problem for Bernoulli random variables, called constrained BL, which has additional constraints incorporated into the BL optimisation problem. These additional constraints ensure the updated expectations are in the range $[0, 1]$; a desirable property for Bernoulli random variables. We show the resulting optimisation problem has an analytical solution.

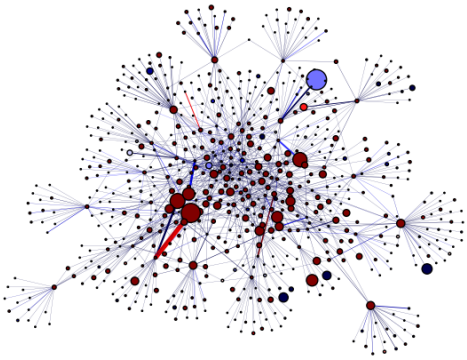
We show how the constrained BL method can be applied to the network-based search problem. For small networks, the updated values using constrained BL are close to the values found using exact inference methods. The performance of the decision problem is not greatly affected by which inference method is used and is robust to changes in the prior. For both simulated data and data from the Enron corpus, the BL model gives comparable performance to the exact inference in the binary MRF model. For networks where the binary MRF model is infeasible, constrained BL gives superior performance to methods that ignore the network structure.



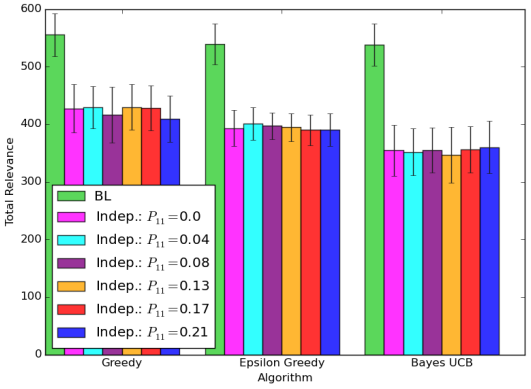
(a) 448 nodes and 630 edges



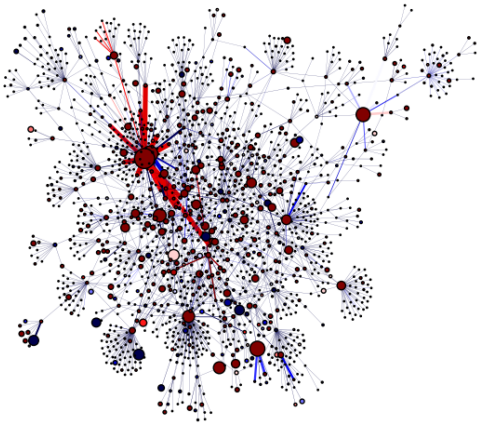
(b) Cumulative Relevance: 448 nodes



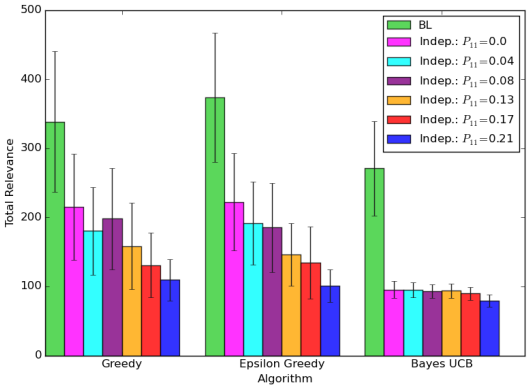
(c) 669 nodes and 1133 edges



(d) Cumulative Relevance: 669 nodes

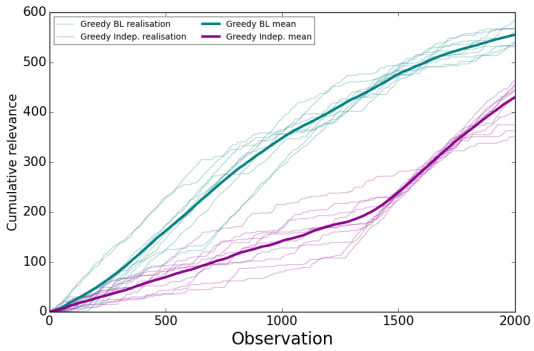


(e) 1641 nodes and 1957 edges

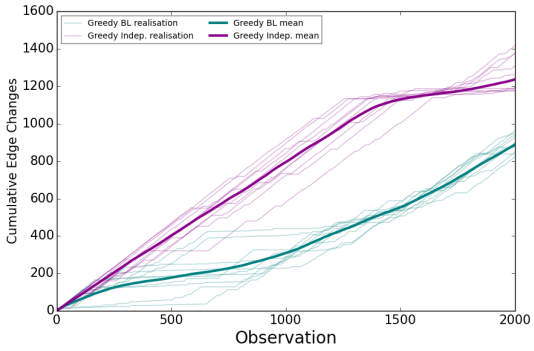


(f) Cumulative Relevance:1641 nodes

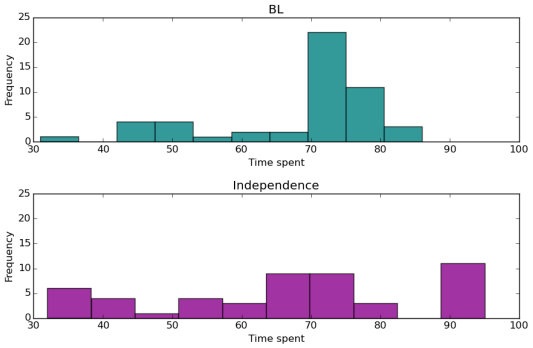
Figure 4.7.2: The mean and ±1 standard error bars for the total number of relevant items found for the greedy, ϵ-greedy and Bayes-UCB policies run on subsets of the Enron Corpus for the BL model and independence models with a range of prior clique factors, chosen to match the mean of the BL model prior. For each network the BL model performs better than the independence model.



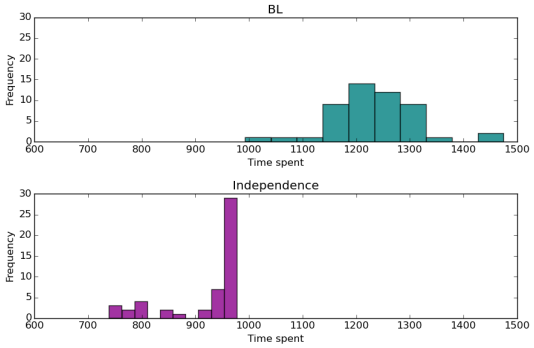
(a) Cumulative relevance for greedy policy



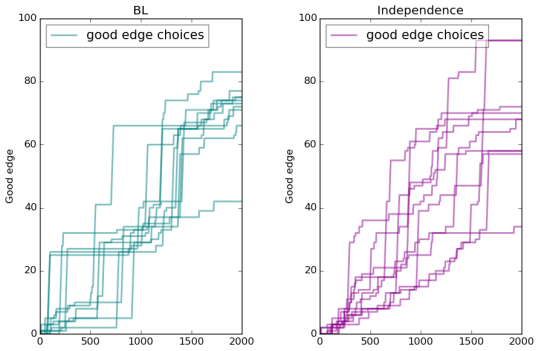
(b) Cumulative number of edge changes for greedy policy



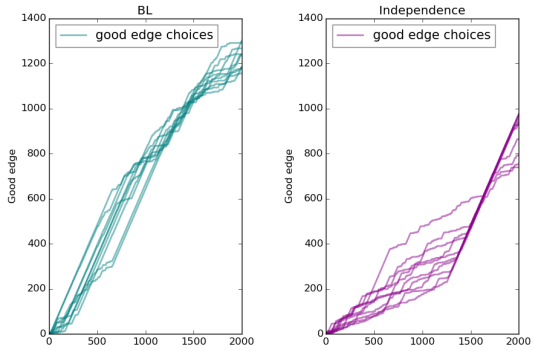
(c) Distribution of time spent on an edge in the top 95th quantile



(d) Distribution of time spent on an edge in the top 90th quantile

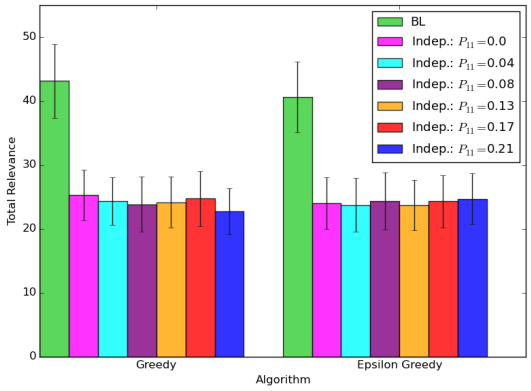


(e) Cumulative number of observations on an edge in the top 95th quantile

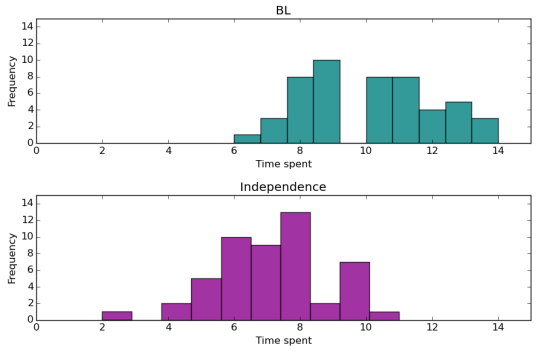


(f) Cumulative number of observations on an edge in the top 90th quantile

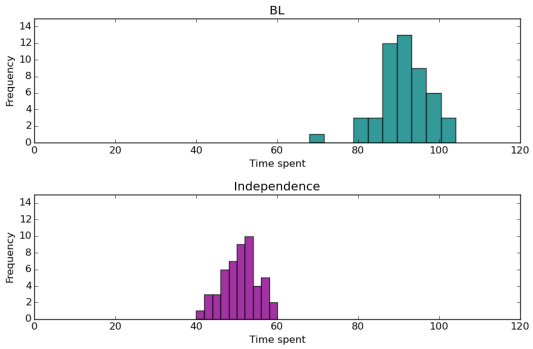
Figure 4.7.3: Results for the greedy policy for the network shown in Figure 4.7.2c and focusing on the independence model with $P_{11} = 0.13$. Figure 4.7.3a shows the cumulative number of relevant items observed. Figure 4.7.3b shows the cumulative number of times the algorithm changed edges. Figure 4.7.3c (4.7.3d) show the number of times items are observed on good edges; where a good edge is one whose probability is above the 95th (90th) quantile of all edge probabilities. For this network, the 95^{th} quantile is 0.5 and the 90^{th} quantile is 0.124. Figure 4.7.3e and 4.7.3f show the results for 10 repetitions of the greedy policy. The results show that the BL model tends to find the good edges faster than the independence model and then screens more items on these edges.



(a) Cumulative Relevance: 669 node



(b) Distribution of times an item on an edge in the top 95th quantile is observed



(c) Distribution of times an item on an edge in the top 90th quantile is observed

Figure 4.7.4: Results for the greedy heuristic for the network shown in Figure 4.7.2c, with 669 nodes and focusing on the independence model with $P_{11} = 0.13$. Figure 4.7.4a shows the mean and ± 1 standard error bars for the total number of relevant items found after 500 observations for the greedy and ϵ-greedy policies. Figure 4.7.4b and 4.7.4c show the number of times items are observed on good edges where a good edge is considered one above the 95th quantile of true probabilities on edges and above the 90th respectively. For this network, 95^{th} quantile is a true probability of 0.5 and the 90^{th} quantile is a true probability of 0.124. The BL model screens far more items on the good edges than the independence model.

Chapter 5

Sequential Monte Carlo with Bayes Linear

5.1 Introduction

In the previous chapter, BL methodology was introduced as a method to approximate inference in a binary MRF model. The BL updates provided an approximation to the posterior expectation and variance for inference in network-based search. In many cases, we showed that these approximations are close to the values found from exact inference in the binary MRF model. However, BL methodology uses a linear approximation to the relationship between the observations and the latent variables. The linear approximation will give a poor representation of highly non-linear relationships, so the BL updates will give a poor approximation. In Section 5.2, we show how the non-linearity in the relationship affects how well the BL model approximates inference in the MRF.

In this chapter, we introduce an alternative approach to inference for sequential search on binary networks to overcome this problem. The method involves combining SMC and BL updates to approximate the joint distribution of the latent variables given observations. Although this method is more computationally expensive than the BL model, it does not

Z_0	Z_1	$P(Z_0, Z_1)$
0	0	0.3
0	1	0.2
1	0	0.2
1	1	0.3

(a) Clique factor

Z_0	Z_1	$E(P_{01}\|Z_0, Z_1)$
0	0	0.1
0	1	0.5
1	0	0.5
1	1	0.9

(b) Conditional 1

Z_0	Z_1	$E(P_{01}\|Z_0, Z_1)$
0	0	0.01
0	1	0.01
1	0	0.01
1	1	0.5

(c) Conditional 2

Z_0	Z_1	$E(P_{01}\|Z_0, Z_1)$
0	0	0.01
0	1	0.75
1	0	0.75
1	1	0.8

(d) Conditional 3

Table 5.2.1: Distributions used to define the MRF and BL models. Table 5.2.1a is the clique factor used to define the binary MRF and the relationship between latent variables and observations is set to one of the conditional probability distributions in Table 5.2.1b to Table 5.2.1d. Conditional 1 in Table 5.2.1b has a linear relationship with the sum of the node relevancies whereas the other two have non-linear relationships.

require a linear approximation to the relationship between the latent variables and the observations.

5.2 Effect of Non-Linearity on the BL Approximation

The BL model relies on a linear approximation between the observations and the latent variables in the MRF. We explore the effect non-linearity has on the BL approximation for a network with a single edge. The true network is taken as the binary MRF model defined using the clique factor in Table 5.2.1a and the BL model is taken as an approximation to this. The difference in posterior expectations and variances for the two models is considered. The relationship between the latent variables and the observations is defined through one of the conditional probability distributions in Tables 5.2.1b to 5.2.1d. We consider fixed conditional probabilities for the purpose of showing the effect of non-linearity on the accuracy of the BL model. Given the value of the random variable at the nodes, the probability of a relevant observation on the edge is fixed.

Conditional 1, in Table 5.2.1b, has a linear relationship with the sum of the node realisations

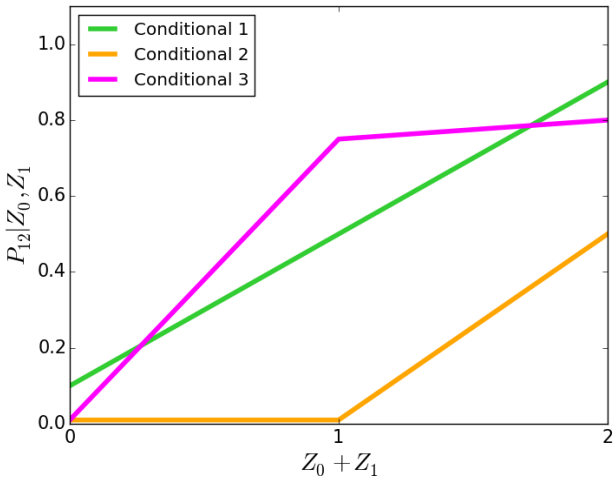
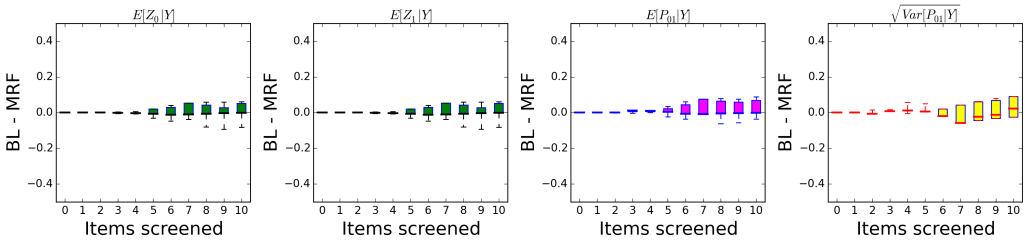


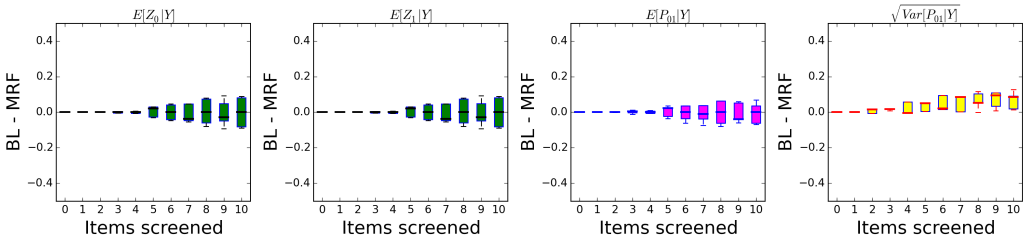
Figure 5.2.1: The conditional probability distributions, in Tables 5.2.1b to 5.2.1d, plotted against the sum of the node values. The relationship is non linear for Conditional 2 and Conditional 3.

$(Z_0 + Z_1)$. This conditional probability can be used to demonstrate how well the BL model approximates the MRF model for a linear relationship between the latent variables and observations. Conditional 2 and Conditional 3 in Tables 5.2.1c and 5.2.1d respectively have non-linear relationships with the sum of the node realisations. Conditional 2 in Table 5.2.1c has high probability of a relevant observation when both node random variables are 1; the other conditional probabilities are low. Conditional 3 in Table 5.2.1d only requires one of the node random variables to be 1 for the conditional probability to be high. The relationship between the sum of latent variables and the probabilities is shown in Figure 5.2.1 for the three conditional probability distributions.

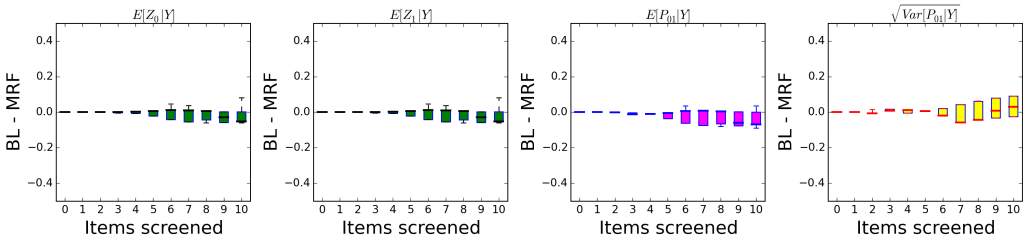
For each conditional probability distribution, and for each possible combination of true node realisations, we simulate 50 sets of 10 observations and compare the distribution of differences between the binary MRF model and the BL model after each observation, over all repetitions. Figure 5.2.2 shows the difference in the updated values for Conditional 1. The BL model gives a good approximation to the linear relationship between latent variables and observations, hence the updated values in the two models are very similar. The mean difference remains around zero, and nearly all observations have a difference of less than 0.1.



(a) $(Z_0, Z_1) = (0, 0)$



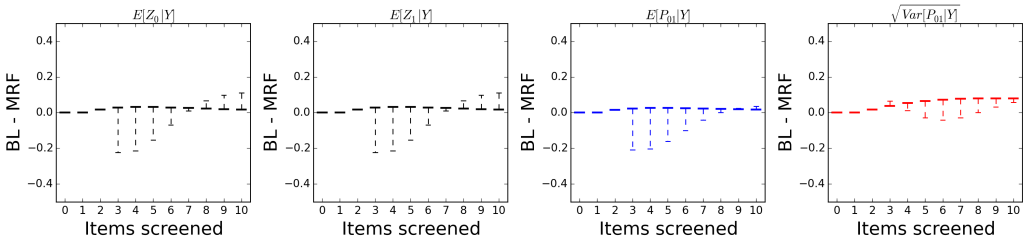
(b) $(Z_0, Z_1) = (1, 0)$



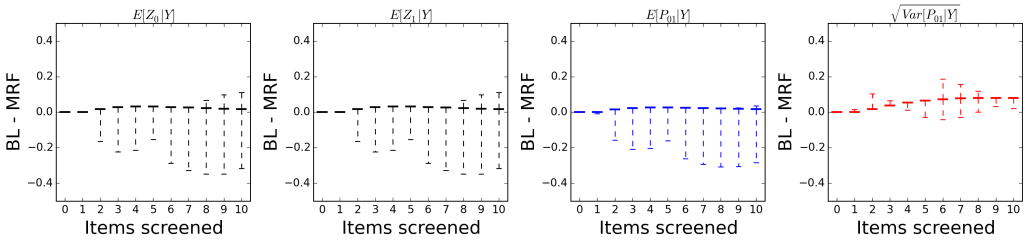
(c) $(Z_0, Z_1) = (1, 1)$

Figure 5.2.2: The distribution of differences over 50 sets of 10 random observations, given the relevance of the nodes, between BL updates and exact inference in the MRF using Conditional 1 to define the relationship between latent variables and observations. The prior conditional distribution is linear in the sum of node relevancies (the latent variables), and hence the BL model gives a good approximation to the MRF model.

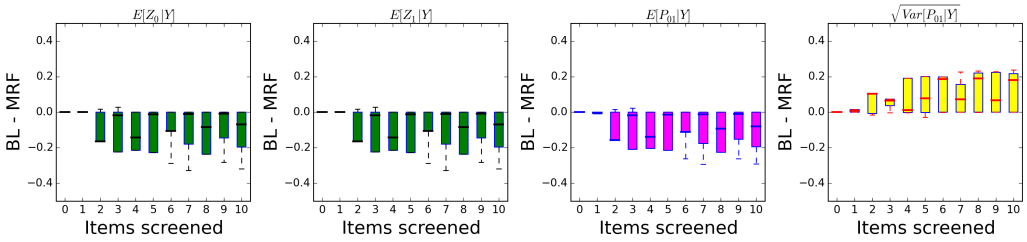
For non-linear relationships, the errors introduced on a single edge are far more noticeable. The distribution of differences using Conditional 2 (Table 5.2.1c), show that for true relevance values $(0, 0)$ and $(0, 1)$, the BL model gives a good approximation to the MRF model, see Figure 5.2.3. These two conditional probabilities dominate the linear approximation. As a result, the BL model poorly approximates the updates when the true relevance is $(1, 1)$ by under estimating the updated expectations. The linear approximation using Conditional 3 (Table 5.2.1d) results in errors in all three true node realisations, see Figure 5.2.4. These differences are more variable than for Conditional 2. For Conditional 2 and Conditional 3, the errors in $Z_0 \mid \mathbf{Y}$ are passed on to errors in $P_{01} \mid \mathbf{Y}$ used in the decision algorithm. The



(a) $(Z_0, Z_1) = (0, 0)$



(b) $(Z_0, Z_1) = (1, 0)$



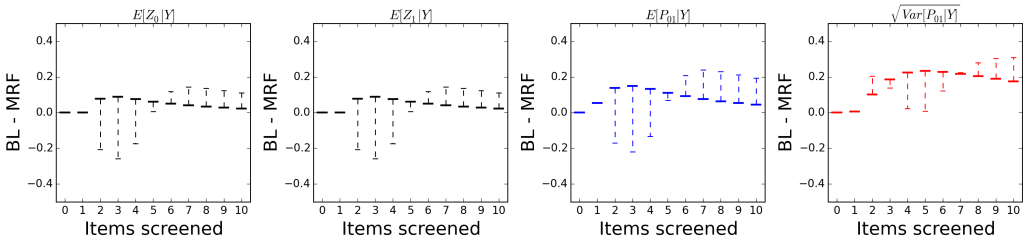
(c) $(Z_0, Z_1) = (1, 1)$

Figure 5.2.3: The distribution of differences over 50 sets of 10 random observations, given the relevance of the nodes, between BL updates and exact inference in the MRF using Conditional 2. When the true relevance of the nodes is $(1, 1)$, the errors induced using BL are large.

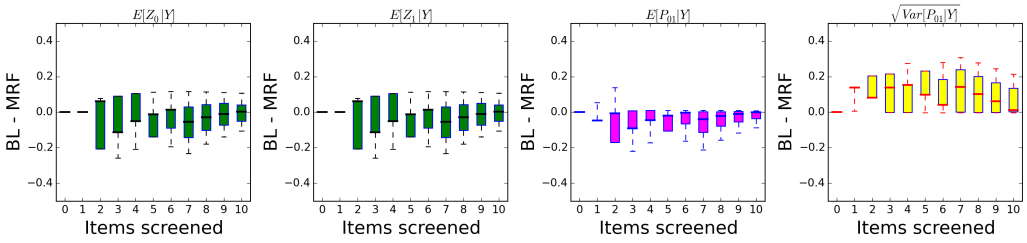
mean difference tends to be close to zero after 10 observations, suggesting the BL model picks up enough of the distribution so that the 10 observations have provided sufficient information to detect the true relevance of the nodes with almost certainty.

5.3 Monte Carlo Sampler for Multivariate Binary Random Variables

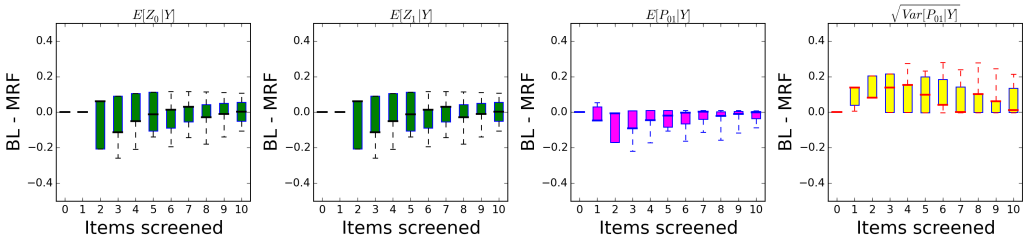
The previous section shows the difficulty encountered when using BL updates if the conditional distribution of the observations is highly non-linear. To overcome this, one approach



(a) $(Z_0, Z_1) = (0, 0)$



(b) $(Z_0, Z_1) = (1, 0)$



(c) $(Z_0, Z_1) = (1, 1)$

Figure 5.2.4: The distribution of differences over 50 sets of 10 random observations, given the relevance of the nodes, between BL updates and exact inference in the MRF using Conditional 3. There is a lot of variability in the differences between the BL model and MRF model.

is to use Monte Carlo methods to obtain samples from the posterior. Given the sequential nature of our application we will focus on SMC procedures.

In order to implement such a SMC procedure we need a computationally efficient procedure to simulate from our prior distribution for the binary latent variables. As for the BL method of Chapter 4 we will assume that our prior is specified through a mean and covariance. Given the success of this, we will base our simulation approach on the BL simulation method described in Section 4.4.2.

Simulating correlated multivariate binary random variables is a common problem which arises in many areas of statistics (George and McCulloch, 1997; Diggle et al., 2002; Rubin-

stein, 1999). As a result, it is a widely studied area, with a variety of possible solutions. These methods include generation from the full probability specification (Kang and Jung, 2001; Lee, 1993; Gange, 1995), which can be computationally challenging and hence is only suitable for low dimensional problems; latent variable models which use the discretisation of a continuous n dimensional vector to define the multivariate Bernoulli random variables (Emrich and Piedmonte, 1991; Modarres, 2011); and conditional means models which are discussed further in Section 5.3.1. We introduce a method based on BL methodology and show that this is analogous to the conditional means model of Qaqish (2003).

5.3.1 Sampling using Bayes Linear Methodology

The prior for $\mathbf{Z} = (Z_1, Z_2, \ldots, Z_n)$ is specified by a prior mean and covariance. We now provide a recap of the idea from Section 4.4.2 for how we can use BL to simulate the binary random variable $\mathbf{Z}$ and give conditions under which these simulated realisations will have a matching mean and covariance. The joint probability of a set of multivariate binary random variables can be split into conditional distributions such that:

$$p(\mathbf{Z}) = p(Z_1)p(Z_2 \mid Z_1) \ldots p(Z_n \mid \mathbf{Z}_{1:n-1}). \tag{5.3.1}$$

Each conditional random variable is a Bernoulli random variable with:

$$p(Z_i \mid \mathbf{Z}_{1:i-1}) = E[Z_i \mid \mathbf{Z}_{1:i-1}]^{Z_i}(1 - E[Z_i \mid \mathbf{Z}_{1:i-1}])^{1-Z_i}. \tag{5.3.2}$$

We can simulate a joint realisation of the multivariate binary random variables by sequentially simulating them from their conditionals, given the value of previously simulated random variables. Suppose they are simulated in numerical order, such that the realisation of Z_1 is simulated first, followed by Z_2 etc. At step i, the conditional expectation of $Z_i \mid \mathbf{z}_{1:i-1}$ can be approximated using the BL updated expectation as:

$$\hat{E}\left[Z_i \mid \mathbf{z}_{1:i-1}\right] = E\left[Z_i\right] + \mathrm{Var}(Z_i, \mathbf{Z}_{1:i-1})\mathrm{Var}(\mathbf{Z}_{1:i-1})^{-1}(\mathbf{z}_{1:i-1} - E\left[\mathbf{Z}_{1:i-1}\right]). \tag{5.3.3}$$

The realisation of Z_i is sampled using this conditional expectation, so that with probability $\hat{E}[Z_i \mid \mathbf{Z}_{1:i-1}]$, $Z_i = 1$. Otherwise $Z_i = 0$. This is equivalent to approximating (5.3.2) by:

$$\hat{p}(Z_i \mid \mathbf{Z}_{1:i-1}) = \hat{E}[Z_i \mid \mathbf{Z}_{1:i-1}]^{Z_i}(1 - \hat{E}[Z_i \mid \mathbf{Z}_{1:i-1}])^{1-Z_i}. \tag{5.3.4}$$

The BL conditional expectation is not constrained to be in $[0, 1]$. If $\hat{E}[Z_i \mid \mathbf{Z}_{1:i-1}] > 1$ then $Z_1 = 1$ and if $\hat{E}[Z_i \mid \mathbf{Z}_{1:i-1}] < 0$ then $Z_i = 0$. This is equivalent to using constrained BL when we are only interested in the conditional expectation of the random variables.

Rather than working directly with the BL updated expectation, we consider the Cholesky decomposition of the covariance matrix and use this to simulate the random variables. Assume the mean of the random variables is $\boldsymbol{\mu}$ and the covariance of $\mathbf{Z}$ is $\Sigma = LL^T$, where L is the Cholesky decomposition of Σ. We can introduce a set of random variables, $\boldsymbol{\epsilon}$, with expectation, $E[\boldsymbol{\epsilon}] = 0$ and variance $\mathrm{Var}(\boldsymbol{\epsilon}) = I$ (the identity matrix) such that:

$$\mathbf{Z} = \boldsymbol{\mu} + L\boldsymbol{\epsilon}, \tag{5.3.5}$$

so that the random variables $\mathbf{Z}$ will have mean $\boldsymbol{\mu}$ and covariance Σ. The lower diagonal form of L gives a natural way of sequentially simulating ϵ_1 to ϵ_n, and once $\epsilon_1, \ldots, \epsilon_i$ have been simulated they can be transformed to give $Z_1, \ldots, Z_i$.

Writing the value of ϵ_i in terms of $\boldsymbol{\epsilon}_{1:i-1}$ and Z_i gives:

$$\epsilon_i = L_{ii}^{-1}\left(Z_i - \mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j\right). \tag{5.3.6}$$

Each Z_i must be either zero or one for the random variables to be Bernoulli. Hence the ϵ_i must also be one of two values, found by substituting $Z_i = 0$ and $Z_i = 1$ into (5.3.6). The probability of these two realisations is fixed so that the expectation of ϵ_i is zero regardless of the previously simulated $\boldsymbol{\epsilon}_{1:i-1}$. This is done by setting:

$$p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j. \tag{5.3.7}$$

To ensure this is a probability, if $p > 1$ then p is set to 1 and if $p < 0$ then p is set to 0. Each ϵ_i is simulated, given previously simulated $\epsilon_{1:i-1}$ from:

$$\epsilon_i = \begin{cases} L_{ii}^{-1}\left(1 - \mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j\right) & \text{with probability } p \\ L_{ii}^{-1}\left(-\mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j\right) & \text{with probability } 1-p \end{cases}, \tag{5.3.8}$$

Algorithm 7 can be used to simulate binary random variables using BL updates, from the prior mean and the Cholesky decomposition of the covariance. Results given in Chapter 6 show that if $p \in [0, 1]$, the simulated $\boldsymbol{\epsilon}$ in Algorithm 7 will have zero mean and the identity matrix for the covariance matrix. Even if this property does not hold, we can still use Algorithm 7 but the mean and covariance of the simulated random variables will no longer be the same as the prior values. When the ordering of the random variables is unknown prior to starting the simulation, we can use the CholeskyBanachiewicz algorithm to expand the Cholesky decomposition to accommodate for the next random variable. The computational cost of calculating the ith row given the first $i-1$ rows of the Cholesky decomposition is $O(i^2)$.

The full Cholesky decomposition can be calculated prior to any random variables being simulated if the order in which they are simulated is known. The computational cost of sampling $(Z_1, \ldots, Z_n)$ is $O(n^3)$, as a result of calculating the Cholesky decomposition of the random variables. For sparse networks the precision matrix of the random variables will be sparse and the cost of calculating the Cholesky decomposition may being much smaller. See Section 5.3.1 for more information.

Conditions on the Prior Mean and Covariance

Prior to starting the simulation, assume we know the random variables will be simulated in numerical order. When the probability, p, in Algorithm 7 is in the range $[0, 1]$ for all $i = 1, \ldots, n$ and all possible realisations of $\mathbf{Z}_{1:i-1}$, the simulated random variables will have a mean $\boldsymbol{\mu}$ and variance Σ, see Chapter 6 for more details. We can place conditions on the prior values so that the probabilities are in this range, no matter what the realisation of

Algorithm 7 Simulating Multivariate Binary Random Variables using Bayes Linear

1. Sample $Z_1 = 1$ with probability $p = \mu_1$ and $Z_1 = 0$ otherwise
2. Set $L_{11} = \sqrt{\Sigma_{11}}$
3. For $i = 2, \ldots, n$:
   (a) Expand the Cholesky decomposition L to include Z_i using the Cholesky-Banachiewicz algorithm.
   (b) Set $p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j$.
      i. If $p \notin [0, 1]$:

$$p = \begin{cases} 1 & \text{if } p > 1 \\ 0 & \text{if } p < 0 \end{cases}.$$

   (c) Sample

$$\epsilon_i = \begin{cases} L_{ii}^{-1}\left(1 - \mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j\right) & \text{with probability } p \\ L_{ii}^{-1}\left(-\mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j\right) & \text{with probability } 1 - p \end{cases}, \tag{5.3.9}$$

   (d) Set:

$$Z_i = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j + L_{ii}\epsilon_i, \tag{5.3.10}$$

the random variables is. In order to give these conditions, we rewrite the the probability in (5.3.7) in terms of $\mathbf{Z}$ rather than $\boldsymbol{\epsilon}$ in Lemma 5.3.1.

Lemma 5.3.1. *For a set of Bernoulli random variables $\mathbf{Z} = (Z_1, \ldots, Z_n)$ simulated in numerical order, with mean μ and covariance $\Sigma = LL^T = (\psi^T\psi)^{-1}$. The probability:*

$$p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j \tag{5.3.11}$$

used to simulate $\epsilon_i|\boldsymbol{\epsilon}_{1:i-1}$ can be written as

$$p = h_0^i + \sum_{k=1}^{i-1} h_k^i Z_k \tag{5.3.12}$$

where:

$$h_0^i = \mu_i - \sum_{k=1}^{i-1}\sum_{j=k}^{i-1} L_{ij}\psi_{jk}\mu_k, \tag{5.3.13}$$

and

$$h_k^i = \sum_{j=k}^{i-1} L_{ij}\psi_{jk}. \tag{5.3.14}$$

Proof. *The relationship in* (5.3.5) *can be rewritten as:*

$$\psi(\mathbf{Z} - \boldsymbol{\mu}) = \boldsymbol{\epsilon}. \tag{5.3.15}$$

Using (5.3.15)*, ϵ_j can be written as:*

$$\epsilon_j = \sum_{k=1}^{j} \psi_{jk}\left(Z_k - \mu_k\right). \tag{5.3.16}$$

Substituting this into (5.3.11)*:*

$$p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\sum_{k=1}^{j} \psi_{jk}\left(Z_k - \mu_k\right). \tag{5.3.17}$$

Rearranging the summation:

$$p = \mu_i - \sum_{k=1}^{i-1}\sum_{j=k}^{i-1} L_{ij}\psi_{jk}\mu_k + \sum_{k=1}^{i-1}\sum_{j=k}^{i-1} L_{ij}\psi_{jk}Z_k, \tag{5.3.18}$$

where if $h_0^i = \mu_i - \sum_{k=1}^{i-1}\sum_{j=k}^{i-1} L_{ij}\psi_{jk}\mu_k$ *and* $h_k^i = \sum_{j=k}^{i-1} L_{ij}\psi_{jk}$ *we get the form of* p *in* (5.3.12). □

If the order that the random variables are simulated in is known prior to starting to simulate them, the conditions in Lemma 5.3.2 ensure the probabilities p in Algorithm 7 will be in the range $[0, 1]$, assuming the random variables are simulated in numerical order.

Lemma 5.3.2. *For a set of Bernoulli random variables,* $\mathbf{Z} = (Z_1, \ldots, Z_n)$*, simulated in numerical order, with mean* $\boldsymbol{\mu}$ *and prior covariance* $\Sigma = LL^T = (\psi^T\psi)^{-1}$*, the probability,* p*, in Lemma 5.3.1 will be in* $[0, 1]$ *if the following conditions hold for* $k = 1 \ldots n$*:*

$$h_0^i + \sum_{h_k^i > 0} h_k^i \leq 1, \tag{5.3.19}$$

$$h_0^i + \sum_{h_k^i < 0} h_k^i \geq 0. \tag{5.3.20}$$

where $h_0^i = \mu_i - \sum_{k=1}^{i-1}\sum_{j=k}^{i-1} L_{ij}\psi_{jk}\mu_k$*, and* $h_k^i = \sum_{j=k}^{i-1} L_{ij}\psi_{jk}$*.*

Proof. *The first condition is found by considering the case where all random variables with a positive* h_k^i*,* $k = 1, \ldots, i-1$ *have a realisation of* $Z_k = 1$ *and the remaining random variables have a realisation of zero. This gives the maximum possible probability. The second constraint is found by considering the case where all random variables with a negative* h_k^i*,* $k \neq i$ *have a realisation of* $Z_k = 1$ *and the remaining random variables have a realisation of zero. This gives the minimum possible probability.* □

When we do not know the order in which the random variables will be simulated, guaranteeing that the probability p is in the range $[0, 1]$ is equivalent to requiring that the conditions hold for every possible permutation of realisation orderings. Lemma 5.3.3 gives the conditions on the prior expectation and covariance for the simulated values to be in $[0, 1]$ for

every possible permutation.

Lemma 5.3.3. *For a set of Bernoulli random variables, Algorithm 7 is used to simulate realisations of* $(Z_{\sigma(1)}, \ldots, Z_{\sigma(n)})$ *with mean* $E[Z_{\sigma(1)}, \ldots, Z_{\sigma(n)}] = \boldsymbol{\mu}$ *and the equivalent permutation of the covariance matrix,* $Var(Z_{\sigma(1)}, \ldots, Z_{\sigma(n)}) = \Sigma_\sigma = LL^T = (\psi^T\psi)^{-1}$*, where* σ *is a permutation of the random variables. The probability, p, in Lemma 5.3.1 will be in* $[0, 1]$ *if the following conditions hold for* $k = 1 \ldots n$ *and every possible permutation:*

$$h_0^i + \sum_{h_k^i > 0} h_k^i \leq 1, \tag{5.3.21}$$

$$h_0^i + \sum_{h_k^i < 0} h_k^i \geq 0. \tag{5.3.22}$$

where $h_0^i = \mu_i - \sum_{k=1}^{i-1} \sum_{j=k}^{i-1} L_{ij}\psi_{jk}\mu_k$*, and* $h_k^i = \sum_{j=k}^{i-1} L_{ij}\psi_{jk}$*.*

Proof. *The proof is similar to the proof for Lemma 5.3.2, where each possible permutation of the random variables and their Cholesky decomposition must be considered, as this is not known prior to starting the simulation, and the Cholesky decomposition will be dependent on this ordering.* □

Using the Prior Precision Matrix

Often, the prior will be specified in terms of the prior mean and inverse covariance matrix (precision matrix), Q. The BL updates can be written in terms of these values directly. Assume we can write the prior precision as $Q = \psi^T\psi$ where ψ is the Cholesky decomposition of Q. We can sequentially simulate the $\mathbf{Z}$s in terms of the Cholesky decomposition of the precision matrix and random variables $\boldsymbol{\epsilon}$. The relationship in (5.3.5) can be rearranged to write ϵ in terms of $\mathbf{Z}$:

$$\psi(\mathbf{Z} - \boldsymbol{\mu}) = \boldsymbol{\epsilon}. \tag{5.3.23}$$

From (5.3.23) the random variable Z_i, in terms of $\mathbf{Z}_{1:i-1}$ and ϵ_i is:

$$Z_i = \mu_i + \psi_{ii}^{-1}\left(\epsilon_i - \sum_{j=1}^{i-1}\psi_{ij}(Z_j - \mu_j)\right). \tag{5.3.24}$$

The epsilon are simulated sequentially, so that their expectation is zero:

$$\epsilon_i = \begin{cases} \psi_{ii}\left(1 - \mu_i + \psi_{ii}^{-1}\sum_{j=1}^{i-1}\psi_{ij}(Z_j - \mu_j)\right) & \text{with probability } p \\ \psi_{ii}\left(-\mu_i + \psi_{ii}^{-1}\sum_{j=1}^{i-1}\psi_{ij}(Z_j - \mu_j)\right) & \text{with probability } 1-p \end{cases}, \tag{5.3.25}$$

where $p = \mu_i - \psi_{ii}^{-1}\sum_{j=1}^{i-1}\psi_{ij}(Z_j - \mu_j)$. If $p > 1$ then p is set to one and if $p < 0$ then p is set to zero. Using the prior precision matrix as opposed to the prior covariance matrix has a computational advantage for sparse networks. In this case, the prior precision of the network for binary MRF will also be sparse (Loh et al., 2012) and hence calculating the Cholesky decomposition may be much faster that $O(n^3)$. These methods cannot be fully utilised unless the ordering is known prior to simulation and the ordering is not important, as we cannot reorder the rows and columns of the matrix.

Link to Conditional Means Models

Whilst we have derived our joint distribution for $\mathbf{Z}$ from BL updates, it turns out that the resulting model is equivalent to the conditional means model by Qaqish (2003). The technique models the conditional probability of Z_i as a linear function of the other components and they can generally be applied to high dimensional problems and is flexible enough to allow for a range of correlation structures. For a mean $\boldsymbol{\mu}$ and covariance Σ, Qaqish (2003) considers the model:

$$E[Z_i|z_1,\ldots,z_{i-1}] = \mu_i + b_i^T(\mathbf{z}_{1:i-1} - \mu_{1:i-1}) = \mu_i + \sum_{j=1}^{i-1} b_{ij}(z_j - \mu_j). \tag{5.3.26}$$

For a given mean, μ, and covariance, Σ, the parameter has a closed form of:

$$b_i^T = \Sigma_{i,1:i-1}(\Sigma_{1:i-1,1:i-1})^{-1}. \tag{5.3.27}$$

Qaqish (2003) looks in detail at the range of correlation matrices for which the mean and covariance of the simulated random variables will match the prior mean and covariance for different models and considers the possibility of permutations in the ordering in which the random variables are sampled to ensure that all conditional expectations are in $[0, 1]$. We consider how these random variables can be simulated if the order is unknown prior to starting the simulation and show that the Cholesky decomposition of the covariance matrix or precision matrix can be used to simulate the random variables.

5.4 Sequential Monte Carlo with Bayes Linear for Graphical Models

We have shown that for non-linear relationships the BL model gives a poor approximation to the relationship between the latent variables and the observations. An alternative method is to take a Monte Carlo approximation. We introduce a SMC method that targets the distribution of the latent variables directly involved in the observations in the graphical model. The method no longer requires a linear approximation to the relationship between latent variables and observations. The remaining latent variables can still approximated using a BL approximation given the Monte Carlo samples.

A challenge with applying SMC to our application is dealing with the potentially high dimension latent variables. Our approach utilises the conditional independence structure in the model. In particular, this means that after processing a set of observations we will only simulate and store realisations of the latent variables that directly affect these observations.

An observation on edge (u, v) is conditionally independent of other observations and all other random variables, given Z_u and Z_v (the random variables of the nodes associated with the edge). Let $\mathbf{Z}_{A_t}$ be the set of all random variables associated all nodes linked to edges involved for which we have observations at time t. Given the set $\mathbf{Z}_{A_t}$, the observations will each be conditionally independent of the other random variables variables in the model. Assuming that $\mathbf{Z}$ is tractable, we can calculate the conditional probability of the random variables not

directly involved in observations, $\mathbf{Z}_{\setminus A_t}$, given those directly involved: $p(\mathbf{Z}_{\setminus A_t} \mid \mathbf{Z}_{A_t})$. The full posterior distribution can be written as:

$$p(\mathbf{Z} \mid \mathbf{Y}_{1:t}) = p(\mathbf{Z}_{A_t}, \mathbf{Z}_{\setminus A_t} \mid \mathbf{Y}_{1:t}) = p(\mathbf{Z}_{\setminus A_t} \mid \mathbf{Z}_{A_t}) p(\mathbf{Z}_{A_t} \mid \mathbf{Y}_{1:t}). \tag{5.4.1}$$

If the posterior distribution of $\mathbf{Z}_{A_t}$ is approximated by a collection of weighted particles, $\left\{\mathbf{z}_{A_t}^{(i)}, w_t^{(i)}\right\}_{i=1}^{N}$, the full conditional distribution can be approximated using the Monte Carlo approximation to $p(\mathbf{Z}_{A_t} \mid \mathbf{Y}_{1:t})$ and the conditional probability $p(\mathbf{Z}_{\setminus A_t} \mid \mathbf{Z}_{A_t})$.

The posterior values are used to decide which observation to make next in the decision problem. After observation $Y_{t+1} = y_{t+1}$, the probability space targeted by the Monte Carlo approximation is expanded to include any latent variables involved with the observation that are not already in the probability space. Details of how this is done are given in Section 5.4.1. Expanding the particle space will lead to the number of particles increasing exponentially with the number of random variables in the Monte Carlo approximation. This is avoided by using a resampling step, described in Section 5.4.2. Our presentation above assumes that the distribution for $\mathbf{Z}$ is tractable. In particular, that it is easy to simulate from the conditional distribution of $\mathbf{Z}_{\setminus A_t}$ given $\mathbf{Z}_{A_t}$. For our application we will use the BL approximation to the distribution of $\mathbf{Z}_{\setminus A_t}$. This is described in Section 5.4.3.

5.4.1 Expanding the Monte Carlo Approximation

As the number of random variables that are directly involved in the observations grows, the probability space of the Monte Carlo approximation must be expanded to incorporate additional random variables. After an observation is made, there are three possible options: $\mathbf{Z}_{A_t}$ remains the same, $\mathbf{Z}_{A_t}$ increases by one random variable or $\mathbf{Z}_{A_t}$ increases by two random variables. The three options involve three different propagation steps to approximate $p(Z_{A_{t+1}}|\mathbf{y}_{1:t})$. However they all involve the same re-weighting step to then approximate $p(Z_{A_{t+1}}|\mathbf{y}_{1:t+1})$. We describe the three different propagation steps first, followed by the common re-weighting step.

Suppose the observation $Y_{t+1} = y_{t+1}$ is made on edge (u, v). The sampling space of the Monte Carlo approximation at time $t+1$ remains the same if both the random variables Z_u and Z_v directly involved in the observation are already in $\mathbf{Z}_{A_t}$. This means $\mathbf{Z}_{A_t} = \mathbf{Z}_{A_{t+1}}$ and the particles are propagated forward to time $t+1$ such that:

$$\left\{\mathbf{z}_{A_{t+1}}^{(i)}, w_{t+1}^{(i)}\right\}_{i=1}^{N} = \left\{\mathbf{z}_{A_t}^{(i)}, w_t^{(i)}\right\}_{i=1}^{N}, \tag{5.4.2}$$

the particle approximation to $p(\mathbf{Z}_{A_{t+1}} \mid \mathbf{y}_{1:t})$.

If one of the random variables $Z_m \in \{Z_u, Z_v\}$ is not in $\mathbf{Z}_{A_t}$ then the random variable is added to the Monte Carlo approximation to give $\mathbf{Z}_{A_{t+1}} = (\mathbf{Z}_{A_t}, Z_m)$; the dimension of the Monte Carlo sample is expanded to include Z_m. Each particle at time t corresponds to two particles in the Monte Carlo approximation at time $t+1$. The particle $\mathbf{z}_{A_t}^{(i)}$ leads to the particles $\mathbf{z}_{A_{t+1}}^{(i_0)} = (\mathbf{z}_{A_t}^{(i)}, 0)$ and $\mathbf{z}_{A_{t+1}}^{(i_1)} = (\mathbf{z}_{A_t}^{(i)}, 1)$. The weights associated with $\mathbf{z}_{A_t}^{(i_k)}$, for $k = 0, 1$, are:

$$w_{t+1}^{(i_k)} = w_t^{(i)} \hat{p}(Z_m = k \mid \mathbf{z}_{A_t}^{(i)}), \tag{5.4.3}$$

where the conditional probability, $\hat{p}(Z_m = k \mid \mathbf{z}_{A_t}^{(i)})$, is approximated using BL updates in Section 5.3.1.

If both the random variables Z_u and Z_v are not in Z_{A_t}, both random variables are added to give $\mathbf{Z}_{A_{t+1}}$. The particle $\mathbf{z}_{A_t}^{(i)}$ corresponds to four particles at time $t+1$: $\mathbf{z}_{A_{t+1}}^{(i_{kl})} = (\mathbf{z}_{A_t}^{(i)}, k, l)$ where $k, l \in \{0, 1\}$. The weight associated with the particle $\mathbf{z}_{A_{t+1}}^{(i_{kl})}$ is:

$$w_{t+1}^{(i_{kl})} = w_t^{(i)} p(Z_u = k \mid \mathbf{z}_{A_{t+1}}^{(i)}) p(Z_v = l \mid \mathbf{z}_{A_{t+1}}^{(i)}, Z_u = k) \tag{5.4.4}$$

where the two conditional probabilities are approximated using BL. This gives a Monte Carlo approximation $\left\{\mathbf{z}_{A_{t+1}}^{(i)}, w_{t+1}^{(i)}\right\}_{i=1}^{N}$ to $p(\mathbf{Z}_{A_{t+1}} \mid \mathbf{y}_{1:t})$.

After any additional random variables which are not in the Monte Carlo approximation at time t have been added, the Monte Carlo approximation is updated to give the posterior

including observation y_{t+1}. The weights associated with the posterior distribution $p(\mathbf{Z}_{A_{t+1}} \mid \mathbf{y}_{1:t+1})$ are:

$$w_{t+1}^{(i)} = w_{t+1}^{(i)} p(y_{t+1} \mid \mathbf{z}_{A_{t+1}}^{(i)}) \;\; i = 1, \ldots, N. \tag{5.4.5}$$

Adding a single random variable to the Monte Carlo approximation doubles the number of particles and adding two quadruples the number of particles. Hence, this will lead to an exponential increase in the number of particles as observations are made. To avoid this, we introduce a maximum number of particles, M, and use resampling to ensure that we use at most M particles to approximate the posterior. If we have $N > M$ particles, the particles are resampled to remove those with low weights.

5.4.2 Resampling Step

We limit the number of particles in the Monte Carlo approximation to M particles. The number of particles grows exponentially as the number of random variables in the particle approximation increases. Hence, a resampling step is used in the algorithm.

For discrete random variables, duplicate particles in the Monte Carlo approximation are irrelevant as the information contained within duplicate particles can be summarised by just one particle. The optimal resampling scheme by Fearnhead and Clifford (2003) minimises the expected squared error loss for approximating a discrete probability mass function of finite support, with fewer support points. The algorithm samples particles above a certain threshold, c, with probability one. If the weight is below the threshold value, the particle is resampled with probability $w_t^{(i)}/c$. The value of c is set to the unique root of:

$$\sum_{j=1}^{N} \min(w_t^{(j)}/c, 1) = M. \tag{5.4.6}$$

The resampling algorithm ensures there are no duplicate particles and is given in Algorithm 8.

Algorithm 8 Optimal Resampling

1. For $w_t^{(i)}$, $i = 1, \ldots, N$, and threshold, c:
   (a) If $w_t^{(i)} > c$, resample particle z_i and set weight $\tilde{w}_t^{(i)} = w_t^{(i)}$.
   (b) If $w_t^{(i)} < c$, with probability $w_t^{(i)}/c$ resample particle z_i and set weight $\tilde{w}_t^{(i)} = c$
2. Reweight particles such that: $w_t^{(i)} = \frac{\tilde{w}_t^{(i)}}{\sum_{j=1}^{N} \tilde{w}_t^{(j)}}$.

5.4.3 SMC with BL Approximations

The SMC method in Section 5.4.1 assumes we can calculate and simulate from $p(\mathbf{Z}_{\setminus A_t} \mid \mathbf{Z}_{A_t})$, which is not possible for large graphical models. Instead, BL methods are used to approximate the distributions in the SMC model. Hence, we work with posterior BL means and variances rather than the full probability distributions.

For each random variable in the Monte Carlo approximation, $Z_k \in \mathbf{Z}_{A_t}$, we can calculate the approximate posterior mean and variance directly from this approximation:

$$\hat{E}[Z_k|\mathbf{y}_{1:t}] = \sum_{i=1}^{N} w_t^{(i)} z_k^{(i)}, \tag{5.4.7}$$

$$\widehat{\mathrm{Var}}[Z_k|\mathbf{y}_{1:t}] = \sum_{i=1}^{N} w_t^{(i)} (z_k^{(i)})^2 - E[Z_k|\mathbf{y}_{1:t}]^2. \tag{5.4.8}$$

The posterior mean and variance for the remaining random variables is found by taking a BL approximation to the latent variables not directly involved in observation, given those in the Monte Carlo approximation. For $Z_k \in \mathbf{Z}_{\setminus A_t}$, the posterior expectation is:

$$\hat{E}[Z_k|\mathbf{y}_{1:t}] = \sum_{i=1}^{N} w_t^{(i)} \hat{E}[Z_k|\mathbf{z}_t^{(i)}], \tag{5.4.9}$$

where $\hat{E}[Z_k|\mathbf{z}_t^{(i)}]$ is the BL updated expectation of $Z_k \mid \mathbf{z}_t^{(i)}$ found using the prior mean and covariance of the random variables. The approximate posterior variance $\widehat{\mathrm{Var}}[Z_k|\mathbf{y}_{1:t}]$:

$$\widehat{\mathrm{Var}}[Z_k|\mathbf{y}_{1:t}] = \hat{E}[Z_k^2|\mathbf{y}_{1:t}] - \hat{E}[Z_k|\mathbf{y}_{1:t}]^2, \tag{5.4.10}$$

can also be approximated using the prior BL conditional expectation and the Monte Carlo approximation, where:

$$\hat{E}[Z_k^2|\mathbf{y}_{1:t}] = \sum_{i=1}^{N} w_t^{(i)} \widehat{\mathrm{Var}}(Z_k|\mathbf{z}_{A_t}^{(i)}) + \sum_{i=1}^{N} w_t^{(i)} \hat{E}[Z_k|\mathbf{z}_{A_t}^{(i)}]^2. \tag{5.4.11}$$

The BL conditional variance does not depend on the realisations of the random variables $\mathbf{Z_{A_t}}$ so:

$$\widehat{\mathrm{Var}}(Z_k|\mathbf{z}_{A_t}^{(i)}) = \widehat{\mathrm{Var}}(Z_k|\mathbf{Z}_{A_t}) = \mathrm{Var}(Z_k) - \mathrm{Cov}(Z_k, \mathbf{Z}_{A_t})\mathrm{Var}(\mathbf{Z}_{A_t})^{-1}\mathrm{Cov}(\mathbf{Z}_{A_t}, Z_k) \tag{5.4.12}$$

Using (5.4.12), the posterior squared expectation in (5.4.11) can be simplified by taking the conditional BL variance outside of the summation:

$$\hat{E}[Z_k^2|\mathbf{y}_{1:t}] = \widehat{\mathrm{Var}}(Z_k|\mathbf{z}_{A_t}) + \sum_{i=1}^{N} w_t^{(i)} \hat{E}[Z_k|\mathbf{z}_t^{(i)}]^2. \tag{5.4.13}$$

Including only the random variables directly involved in the observations allows non linearity in this relationship to be captured, whilst minimising the computational cost. The algorithm for SMC with BL is shown in Algorithm 9.

If Lemma 5.3.3 holds, the particles in the Monte Carlo approximation will have a posterior mean and covariance which do not depend on the ordering that the random variables are simulated in. More information on the invariance properties of the BL simulation method is given in Chapter 6.

5.5 Results

In this section, we analyse the SMC model for network-based search. Firstly, we illustrate that for a small network the MRF model and SMC model give very similar posterior values for a non-linear conditional probability distribution, whereas the BL model gives inaccurate estimations. We show how the improved accuracy in the posterior values affects the performance of the decision algorithms for the small network. For a network where resampling

Algorithm 9 SMC with BL for Network-Based Search

Initialisation:

1. Observe Y_1 on edge (u_1, v_1).
2. Let $Z_{A_1} = \{Z_{u_1}, Z_{v_1}\}$ and $\left\{z_1^{(i)}, w_1^{(i)}\right\}_{i=1}^{4}$ be the four possible realisations of the random variables in Z_{A_1} associated with the observation y_1. The weights $w_1^{(i)} = p(\mathbf{z}_1^{(i)})$ are calculated using (5.3.2).
3. Update weights given the observation, $Y_1 = y_1$. For $i = 1, \ldots, 4$:

$$w_1^{(i)} \propto w_1^{(i)} p(Y_1 = y_1 \mid \mathbf{z}_1^{(i)}) \tag{5.4.14}$$

Iterate: For $t = 2, \ldots, T$:

1. Calculate $E[\mathbf{Z}|\mathbf{y}_{1:t-1}]$ and $\mathrm{Var}(\mathbf{Z}|\mathbf{y}_{1:t-1})$, see Section 5.4.3, and use these to decide where to make the next observation.
2. Observe Y_t on edge (u_t, v_t).
3. For $Z_m \in \{Z_{u_t}, Z_{v_t}\} \notin \mathbf{Z}_{A_{t-1}}$, expand particle approximation to incorporate the random variable, see Section 5.4.1, to give $\left\{z_{A_t}^{(i)}, w_t^{(i)}\right\}_{i=1}^{N}$.
4. Update weights $w_t^{(i)} \propto w_t^{(i)} p(y_t|\mathbf{z}_t^{(i)})$.
5. If the number of particles is greater than the maximum sample size, M, then resample the particles, see Section 5.4.2.

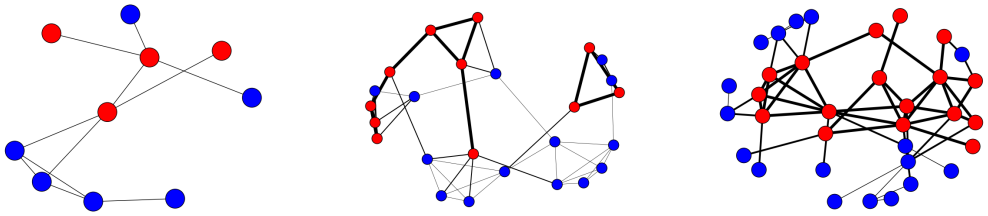
is required, we show that the SMC model with optimal resampling gives superior estimates of the MRF compared to more simple approximations, see Section 5.5.2. A comparison of the computational time for the BL model, SMC model and MRF model is given in Section 5.5.3. Whilst the computational time taken to run one iteration of the decision policy only grows polynomially for the BL and SMC model, we show the computational time for the MRF model increases exponentially, as the number of edges in the network increases.

In Section 5.5.4 we demonstrate the effect of sample impoverishment on the approximation for the Tanzania network by varying the maximum number of particles in the approximation. The effect this sample impoverishment has on the decision algorithm is analysed. For larger networks, we see that the maximum number of particles in the approximation affects the performance of the network search. Finally, we look at the performance of the SMC model compared with the BL model for the Enron data. In Chapter 4, the BL model performed very well in the decision algorithm for this data. We show that the shape of the non-linear distribution may affect how well the BL model does in the decision algorithms when compared to the SMC model. However, for more realistic prior conditional probability distributions, the BL model does just as well as the SMC model at lower computational cost. Some insight on why this might be the case is provided.

5.5.1 SMC on a 10 Node Network

In this section, we show that for a small network with a non-linear prior conditional distribution, the updates in the MRF model and SMC model are very similar, whilst the BL model gives different updates. However, in Section 5.5.1 we show that even with these differences all three models perform very similarly in the decision process.

For networks where the number of nodes is small enough that resampling is not required, updates using the MRF and SMC models will be very similar; all possible realisations of the random variables can be considered in the SMC model. Any errors induced will be a result of the limited amount of prior information assumed in the SMC model, which is used to simulate the particles in the SMC approximation. To demonstrate this, we consider a



(a) 10 node network (b) 25 node network (c) Tanzania network

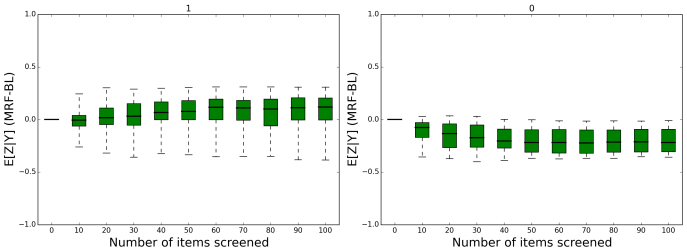
Figure 5.5.1: Networks used to test the SMC model.

network with 10 nodes, shown in Figure 5.5.1a. The prior clique factor in Table 5.5.1a with $[\lambda_1, \lambda_2] = [0.5, 0.5]$ is used to define the prior MRF model of the latent variables; the prior mean and covariance are calculated from the MRF model. The true nodes relevancies are a realisation found using Algorithm 7. The red nodes have a true value of 1 and the blue nodes a true value of 0 in Figure 5.5.1a. The outcome of each observation is sampled from a Bernoulli distribution with a success probability defined using prior conditional A (Table 5.5.1b).

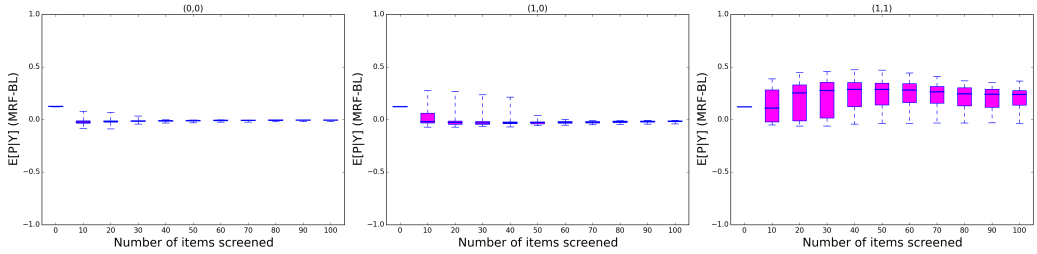
For the same 50 sets of 100 observations on randomly selected edges we run the MRF, BL and SMC models. Figure 5.5.2 shows the difference in expected value between the two approximate methods and the MRF model. These differences are calculated over all repetitions and nodes or edges, after a given number of observations. The errors in the updated values of the SMC model, Figure 5.5.2c and 5.5.2d, are virtually non-existent. For this network and prior conditional probability, the BL model is conservative with its estimation of the effect of an observation on the latent variables; the BL approximation under estimates the posterior expectation when $Z_i = 1$ and over estimates the posterior expectation when $Z_i = 0$, see Figure 5.5.2a.

Decision Problem for Small Network

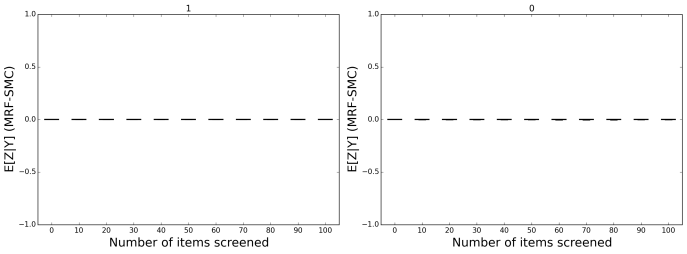
Following the improvement in the approximation using the SMC model compared to the BL model for the small network we would like a superior performance in the network-based search algorithm to find relevant observations. Throughout this chapter, we consider



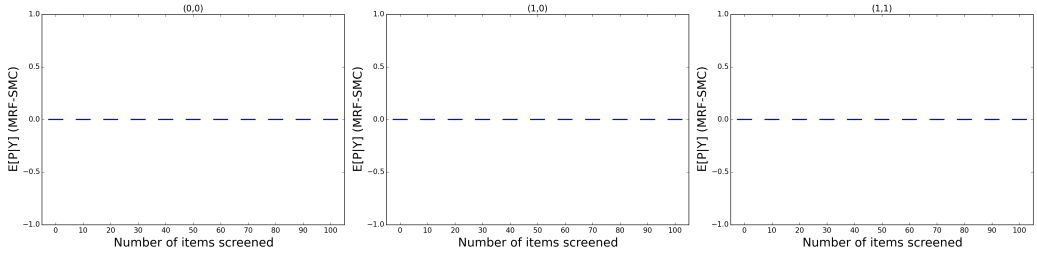
(a) BL: $E(Z|Y)$



(b) BL: $E(P|Y)$



(c) SMC: $E(Z|Y)$



(d) SMC: $E(P|Y)$

Figure 5.5.2: Small network results: the distribution of differences over 50 sets of random observations, given the relevance of the nodes, between Bayes Linear updates and exact inference in the MRF model using prior conditional A, in Table 5.5.1b.

Z_i	Z_j	$P(Z_i, Z_j)$
0	0	$1 + \lambda_1$
0	1	1
1	0	1
1	1	$1 + \lambda_2$

(a) Clique factor

Z_i	Z_j	$\alpha(Z_i, Z_j)$	$\beta(Z_i, Z_j)$	$E[P_{ij}\|Z_i, Z_j]$
0	0	0.1	10.0	0.01
0	1	0.1	10.0	0.01
1	0	0.1	10.0	0.01
1	1	1.0	1.0	0.5

(b) Prior conditional A

Z_i	Z_j	$\alpha(Z_i, Z_j)$	$\beta(Z_i, Z_j)$	$E[P_{ij}\|Z_i, Z_j]$
0	0	1.0	9.0	0.1
0	1	3.0	1.0	0.75
1	0	3.0	1.0	0.75
1	1	4.0	1.0	0.8

(c) Prior conditional B

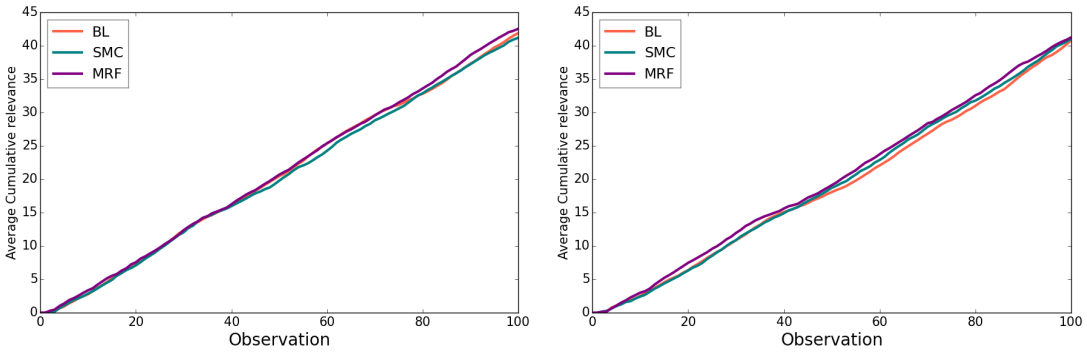
Table 5.5.1: Prior conditional probability distributions and prior clique factor used to test the SMC model and BL model for a non-linear relationship between observations and latent variables.

the greedy and Bayes-UCB heuristics used in Chapter 4. The greedy policy consistently performed well whilst Bayes-UCB considers both the posterior mean and variance in making the decisions. We run the screening process using the two decision policies for 50 sets of 100 observations using the MRF, SMC and BL models and compare the mean cumulative number of relevant items found in the screening process, see Figure 5.5.3a and 5.5.3b.

The mean cumulative relevance shown in Figure 5.5.3 is nearly the same for all model and both decision policies. The better approximation of the SMC model for this network does not affect the performance of the screening algorithm; the BL model may give a good enough approximation to the latent variables to perform well in the decision algorithms even for non-linear relationships. For such a small network, all methods will quickly learn the relevance of the nodes, even in the BL model.

5.5.2 A Comparison with Simpler Approximation Methods

In this section, we justify the use of the SMC sample as an approximation to the MRF model by comparing the updates with other simpler approximate inference methods. The keys ideas behind the SMC method are:



(a) Greedy (b) Bayes-UCB

Figure 5.5.3: The mean cumulative number of relevant items screened using the three models on the small network. The true network is defined using non-linear prior conditional A (Table 5.5.1b) and the clique factor (Table 5.5.1a) with $[\lambda_1, \lambda_2] = [0.5, 0.5]$. The BL model finds just as many relevant items as the SMC and MRF models.

1. It only takes a Monte Carlo sample of the random variables directly involved in the observations.

2. It uses BL updates to approximate both the probability of the realisations within the Monte Carlo sample and the latent variables not in the sample, given the realisations.

3. We have chosen the optimal resampling scheme.

The simplest approximation technique we consider is an IS method, where a Monte Carlo sample is taken to the full set of latent variables, given the prior mean and covariance. This allows us to evaluate the impact of only including a subset of the latent variables in the approximation. The particles are sampled using Algorithm 7 and corresponding weights are then calculated using BL updates. As observations are made, we then update the weights to approximate the posterior distribution of $\mathbf{Z} \mid \mathbf{Y}$. Henceforth, this method is called the BL IS sampler.

We can evaluate the impact of using BL updates to approximate the probabilities in the Monte Carlo sample by also considering an IS sampler where the samples are re-weighted so they approximate the true posterior distribution, hence forth called the MRF IS sampler.

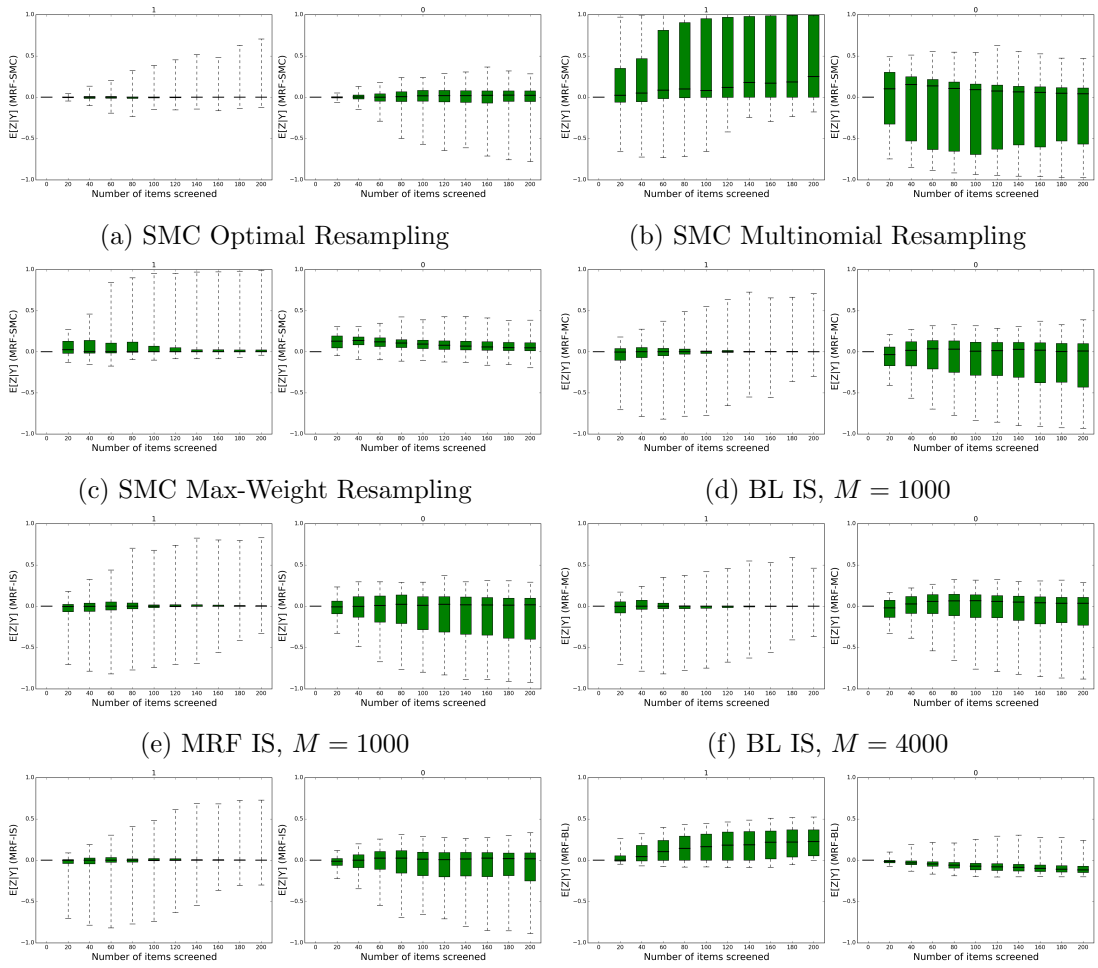
The optimal resampling scheme given in Algorithm 8 is used to decide which particles to include in the Monte Carlo approximation. A key feature of this resampling algorithm is

that no duplicate particles are selected. These particles are chosen to minimise the squared error loss for approximating a discrete probability mass function with a finite support. We compare the method to the default method for resampling in particle filters, multinomial resampling, which allows for duplicate particles in the Monte Carlo sample. Additionally, we compare the method with max-weight resampling (Tugnait, 1982; Punskaya et al., 2002), which ensures no duplicate particles are selected but deterministically selects the particles with maximum weight.

Finally, we also compare the updates using the BL model described in Chapter 4. The SMC method was developed to better cope with non-linearities in the relationship between observations and latent variables. Hence, for the conditional distributions considered, we would expect this to give a worse approximation to the MRF model.

We run each approximate inference method and the MRF model on 50 sets of 200 observations screened on randomly chosen edges for a network with 25 nodes, shown in Figure 5.5.1b. This network is large enough to require resampling in the SMC model. The true node realisations are simulated using the method in Section 4.6.2, with $\delta = 0.8$. The prior clique factor in Table 5.5.1a with $[\lambda_1, \lambda_2] = [0.4, 0.4]$ is used to define the prior MRF model for the $\mathbf{Z}$s. The values for λ_1 and λ_2 are chosen to ensure that the conditions in Lemma 5.3.3 are met. The prior mean and covariance used in the approximate inference methods are calculated directly from the MRF model. The relationship between the latent variables and observations is described using the beta distribution with parameters given by prior conditional A in Table 5.5.1b.

The differences between the posterior expectation of the latent variables using the approximate methods and MRF model are illustrated after a given number of observations and over all repetitions, see Figure 5.5.4. For the SMC methods and the IS methods, we consider the maximum number of particles, $M = 1000$. Additionally, we run the IS methods with $M = 4000$, which ensures the run time updating the Monte Carlo sample in the IS models given an observation roughly matches the maximum run time of an iteration in the SMC method, as the IS methods are less computationally expensive. The plots are split over the



(a) SMC Optimal Resampling

(b) SMC Multinomial Resampling

(c) SMC Max-Weight Resampling

(d) BL IS, $M = 1000$

(e) MRF IS, $M = 1000$

(f) BL IS, $M = 4000$

(g) MRF IS, $M = 4000$

(h) Bayes Linear

Figure 5.5.4: 25 node network with observations on random edges. The distribution of differences between inference in the Markov random field using prior conditional A and the approximate inference methods over 50 sets of random observations, given the relevance of the nodes. The Monte Carlo methods have a maximum number of particles of $M = 1000$. The SMC models give the best approximation to the MRF model for this network.

true relevance of the nodes in the network.

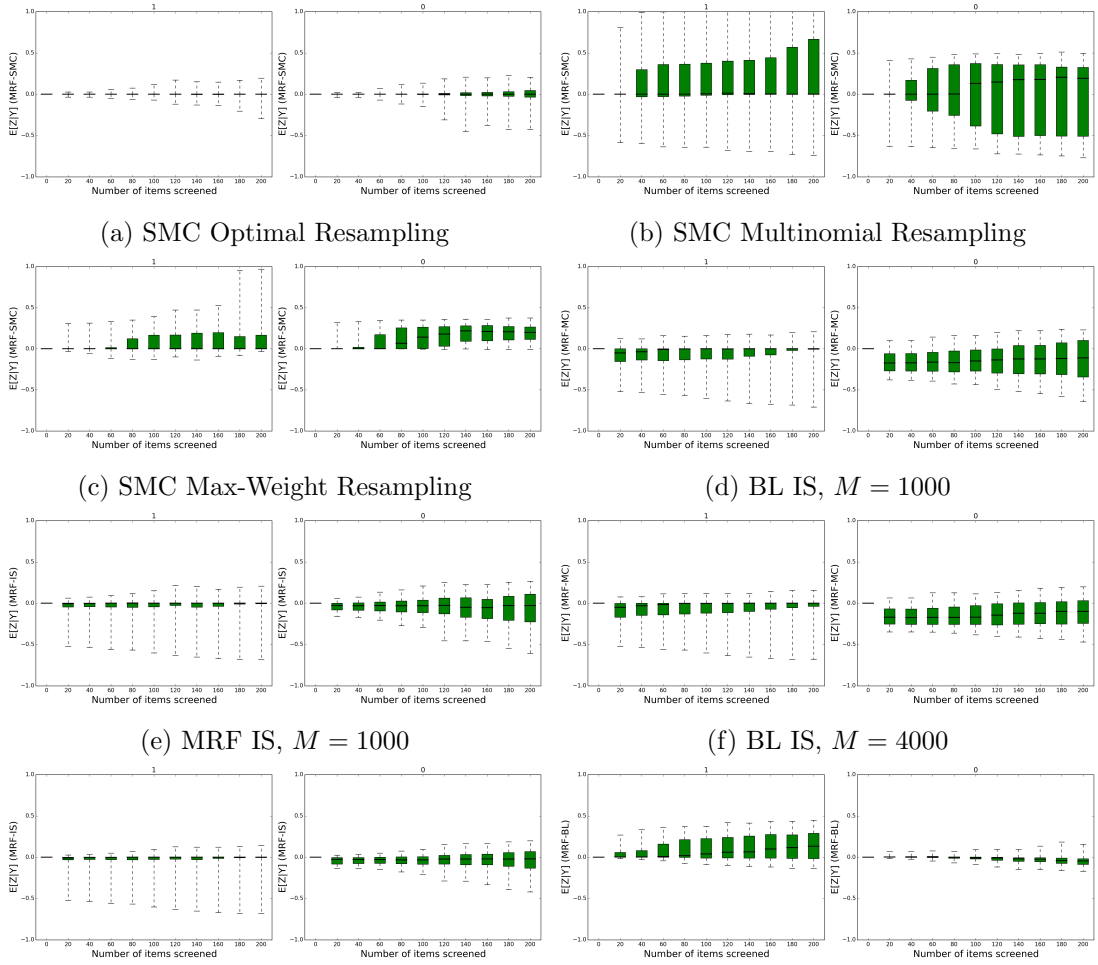
The SMC methods with optimal and max-weight resampling schemes produce the smallest errors compared to the MRF model. The results for optimal resampling in Figure 5.5.4a provide a slightly better approximation compared to max weight resampling, particularly for nodes with a true relevance of 1. SMC using multinomial resampling gives a very poor approximation. The BL IS and MRF IS models perform poorly in comparison to the SMC method with optimal resampling, even when we consider a larger number of particles. The

particles in the IS methods depend only on the prior mean and covariance, whilst the SMC also takes into consideration the observations when expanding the particle space and possibly only contains a subset of the random variables. The BL IS model performs similarly to the MRF IS model, suggesting that the BL method gives a good approximation to the probabilities in the particle approximation. The BL model gives a poor approximation when the true node relevance is one. The prior conditional probability distribution is non-linear so this is to be expected.

The SMC model performs well when observations are on random edges compared to the other simpler approximate methods. However, the SMC model was developed for inference within sequential decision problems where there is likely to be less exploration within the network than as a result of choosing random edges. We consider the set of observations from running the greedy algorithm 50 times on the MRF model rather than random edge observations. The greedy algorithm has far less exploration and so it is possible that observations are made on only a small number of edges in the network. Figure 5.5.5 shows the difference in posterior values in the MRF and the approximate inference methods. The SMC model with optimal resampling gives the most accurate approximation to the MRF model, see Figure 5.5.5a. Max-weight resampling gives a less accurate approximation than using optimal resampling; the diversity in the particles is lost at an earlier point in time than with the optimal resampling. This shows the influence a good resampling scheme can have on the accuracy of the method.

5.5.3 Comparison of Computational Time

In this section, we look at the improvement in computational time from using the approximate inference methods compared with the MRF model, which uses the junction tree algorithm for exact inference. We show that as the size and connectivity of the network increases, the computational time required for the BL and SMC methods are far smaller than the MRF model. The computational cost of exact inference in the MRF model grows exponentially with the tree with of the graph, whilst the BL and SMC methods grow poly-



(a) SMC Optimal Resampling

(b) SMC Multinomial Resampling

(c) SMC Max-Weight Resampling

(d) BL IS, $M = 1000$

(e) MRF IS, $M = 1000$

(f) BL IS, $M = 4000$

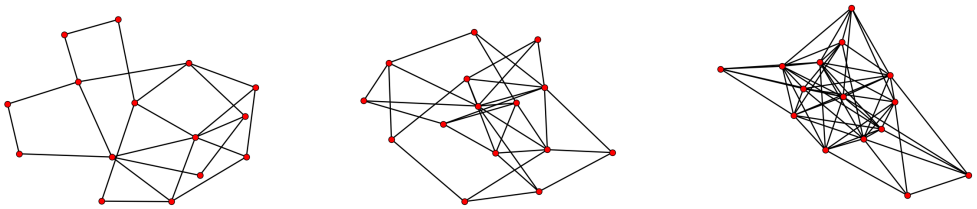
(g) MRF IS, $M = 4000$

(h) Bayes Linear

Figure 5.5.5: 25 node network with observations sampled using the greedy heuristic in the MRF model. The distribution of differences between inference in the Markov random field using prior conditional A and the approximate inference methods over 50 sets of random observations, given the relevance of the nodes. The SMC models give the best approximation to the MRF model for this network.

nomially with the number of nodes in the network.

The improvement in computational time will be dependent on the type of network the algorithms are run on. The MRF model using the junction tree algorithm will be faster for sparse networks, where the nodes form in small clusters. These networks will have a small tree width. Whilst for networks with no defined clusters, or dense networks, the networks will have a large tree width. Hence, to compare the computational efficiency, we simulate small networks using two different methods, where the number of nodes and connectivity



(a) 15 nodes, p = 0.1 (b) 15 nodes, p = 0.25 (c) 15 nodes, p = 0.5

Figure 5.5.6: Networks with 15 nodes where each node is connected with probability p to each other node in the network.

can be controlled.

For each combination of parameters used in the methods, 10 networks are simulated. The true node realisations for each network are simulated using the method in Section 4.6.1, with $\rho = 0.5$. The prior clique factor in Table 5.5.1a with $[\lambda_1, \lambda_2] = [0.5, 0.5]$ is used to define the prior MRF model for the $\mathbf{Z}$s. The prior mean is set to 0.5 and the prior covariance used in the BL and SMC methods is defined using the method in Section 4.6.2 with $\rho = 0.8$. The prior conditional beta distribution is given in Table 5.5.1b.

On each of the networks, the greedy decision algorithm is run for 100 iterations. We compare the average computational time taken for a single iteration of the decision algorithm (over the 100 iterations on the 10 networks).

The first method we use to simulate the networks has two parameters: the number of nodes, n, in the network and the probability p that one node is connected to another node. A higher probability of being connected will result in a more dense network. Figure 5.5.6 shows example networks with $n = 15$ nodes and probabilities $p = (0.1, 0.25, 0.5)$.

Figure 5.5.7 shows the log computational time in seconds for between 10 and 30 nodes and a probability of each node being connected set to $p = (0.1, 0.25, 0.5)$. With only 10 nodes, the computationally time to perform an iteration of the decision algorithm in the MRF model is less then the other two models. However, as the number of nodes increases the two approximate inference methods become more computationally efficient than the MRF model. For higher probabilities, p, the computational time for a single iteration of the

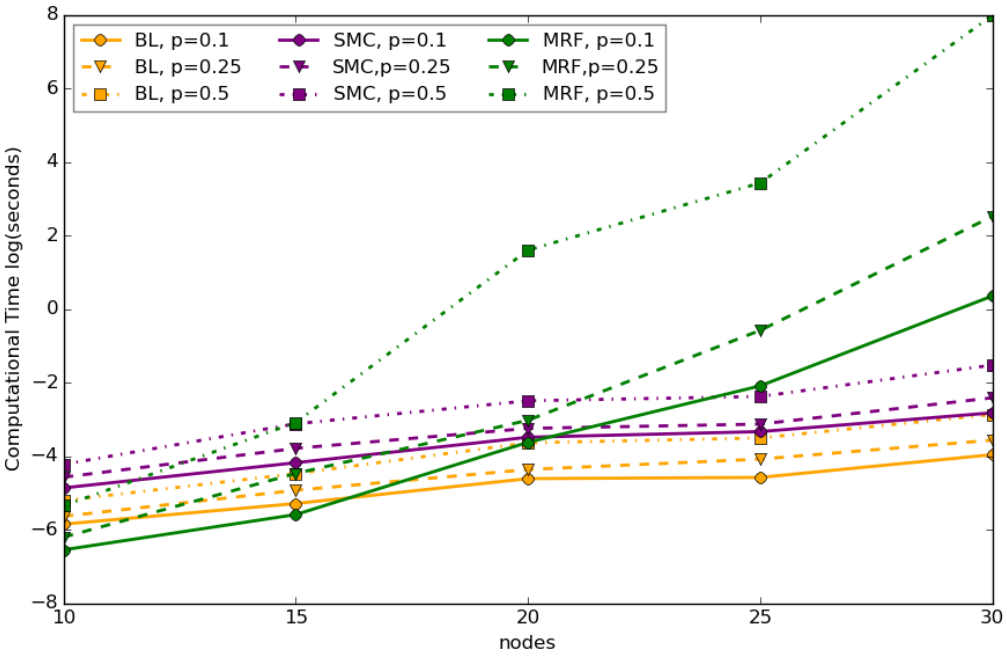
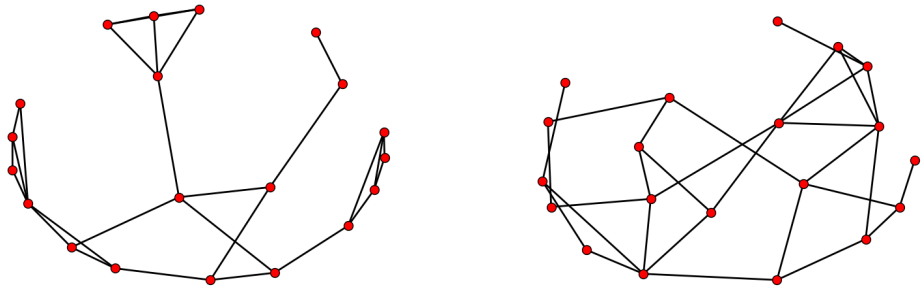


Figure 5.5.7: The log computational time in seconds for a single iteration of the greedy decision algorithm averaged over 10 networks with 100 iterations, for networks with n nodes and probability p that a node is connected to another node.

MRF model increases exponentially as the maximum tree width will soon be similar to the number of nodes in the network. The computational time for the BL and SMC models does not depend on the tree width. They increase polynomially with the number of edges in the network. The computational time of the SMC model is always slightly longer than the BL model.

The junction tree algorithm used in the MRF model will be more computationally efficient when the nodes in the network group together in small clusters. For networks simulated using the first method, this property is not taken into consideration. The second method we use to simulate networks is the `relaxed_caveman_graph` function in NetworkX (Hagberg et al., 2008) which has 3 parameters: The number of cliques, I; the number of nodes, k, in each clique; and the probability each node is re-wired, q. The junction tree algorithm is more suited to networks when q is low. The model can split the network up into small cliques. Figure 5.5.8 shows a network with 5 cliques and with 4 nodes in each clique where the probability of rewire is 0.2 and 0.6.



(a) I = 5, k = 4, q = 0.2 (b) I = 5, k = 4, q = 0.6

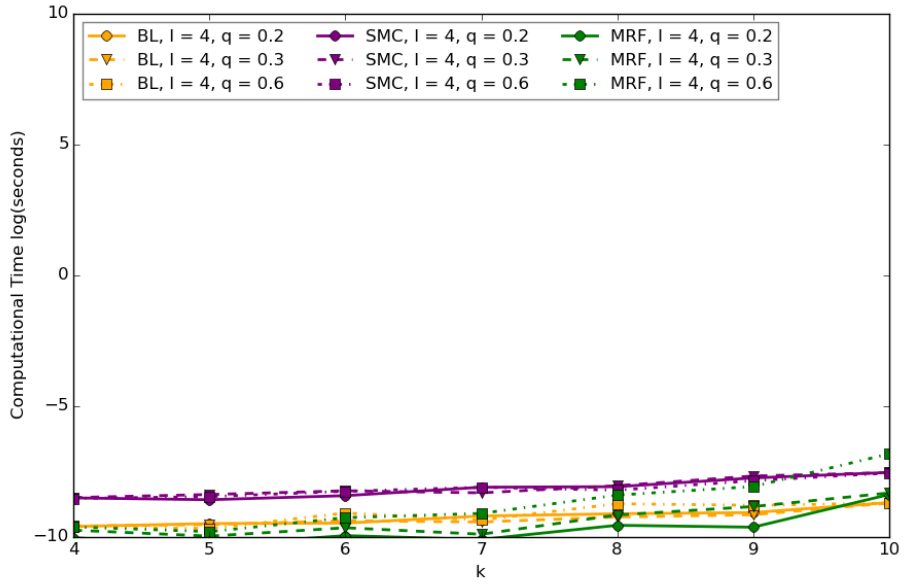
Figure 5.5.8: Networks simulated using the `relaxed_caveman_graph` function.

Figure 5.5.9 shows the log computational time in seconds to run a single iteration of the greedy algorithm for networks simulated using the `relaxed_caveman_graph` function. Figure 5.5.9b shows the results with $I = 4$ nodes in each clique and Figure 5.5.9b shows the results with $I = 8$ nodes in each clique. With 4 nodes in each clique, the SMC model has the highest average computational time for nearly all values of k, apart from $k = 10$. In the case where $k = 10$ and $q = 0.3$ the computational time of the MRF model has increased above the other two models.

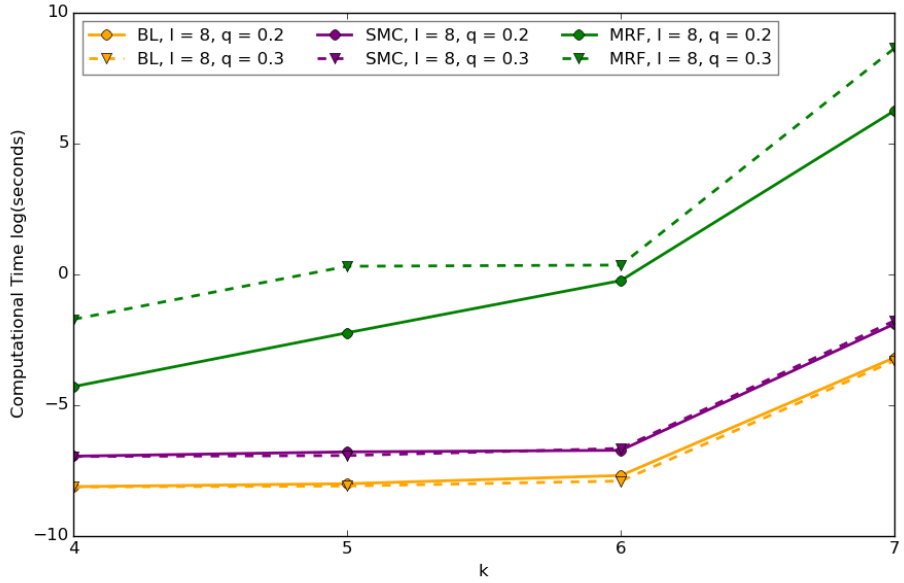
The BL and SMC methods do not depend on the probability that each edge is re-wired. However, the probability of re-wire does effect the computational time of the MRF model. With a small number of cliques, the MRF model is the most computationally efficient but as the number of nodes increases the computational cost eventually rises above the BL model. For higher re-wire probabilities, this increase in computational cost increases at a faster rate as the tree width of the graph increases. For 8 nodes in each clique, the computational time for the BL and SMC methods remains short whilst the computational time for the MRF model quickly increases as the number of cliques increases.

5.5.4 Sample Impoverishment in Larger Networks

For larger networks, we look at how the maximum number of particles in the Monte Carlo approximation, M, affects the accuracy of the SMC method, and how this accuracy in the



(a) k = 4



(b) k = 8

Figure 5.5.9: The log computational time in seconds for a single iteration of the greedy decision algorithm averaged over 10 networks with 100 iterations on each network. The networks simulated with I cliques, k nodes in each clique and probability q that each node is re-wired.

updates affects the decision policy. In particular, we look at the effect of only having a small M on sample impoverishment.

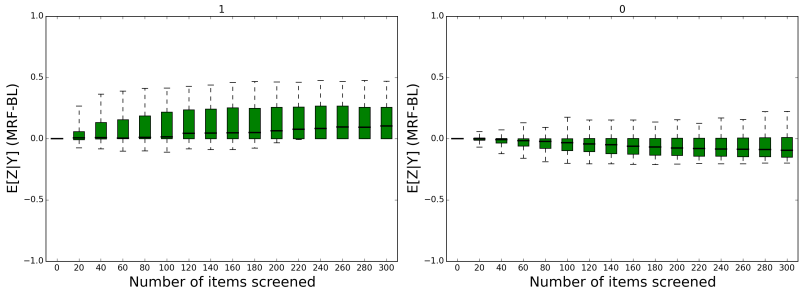
The more particles there are in the Monte Carlo approximation, the more accurate the up-

dates will be, but the more computationally intensive the method will be. For the Tanzania network, shown in Figure 5.5.1c, the prior models are defined using the clique factor in Table 5.5.1a, with $[\lambda_1, \lambda_2] = [0.4, 0.4]$ and prior conditional A. We run updates using the SMC algorithm with a varying number of maximum particles. When M is $10,000$ or $100,000$, the errors compared to updates in the MRF are very small, see Figure 5.5.10c and 5.5.10d. However, when M is reduced to $1,000$, there are examples of sample impoverishment. The difference between the MRF model and the SMC model with $1,000$ particles is sometimes one, suggesting that at an earlier step in the resampling scheme all the particles with realisation one have been removed for certain random variables. This is likely to happen more often when the prior conditional probability has either very low or very high probabilities. We try to minimise the amount of sample impoverishment by only sampling the random variables directly involved with the observations and using a resampling scheme where no duplicate realisations are allowed. Even when the SMC method gives a poor approximation to a couple of random variables, the majority of errors are very small with $1,000$ particles.

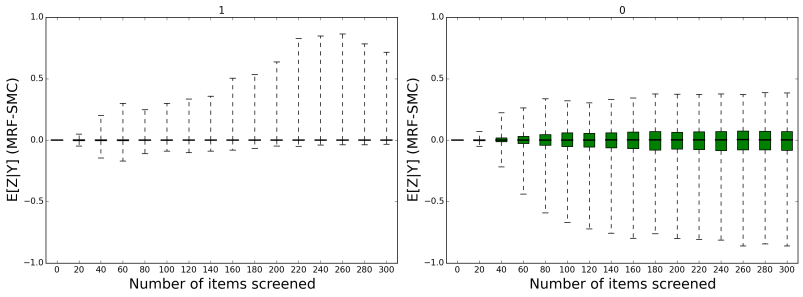
We run the decision algorithms with the SMC models and the BL model. All provide very similar results. The SMC model only finds a few more relevant items on average after 300 observations, see Figures 5.5.11, even when $M = 1,000$. The greedy algorithm using SMC models finds roughly the same number of items as the MRF model. Using the Bayes-UCB heuristic, the approximate models all perform worse than the MRF model.

Caveman Networks

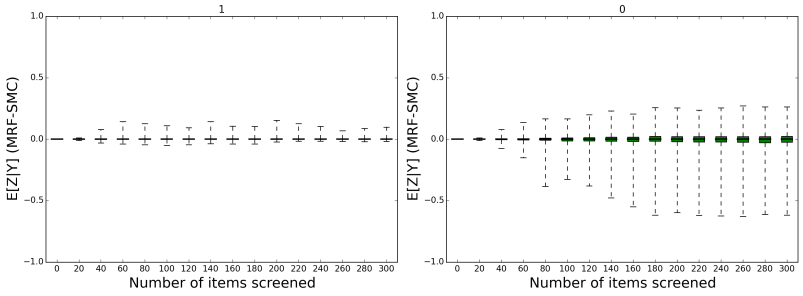
For larger networks, we start to see the effect of sample impoverishment in the SMC sampler on the decision problem. We simulate three networks using `relaxed_caveman_graph` function in NetworkX (Hagberg et al., 2008) and simulate the relevance of the nodes using the method in Section 4.6.1 with $\rho = 0.8$. The networks are shown in Figure 5.5.12a to 5.5.12c and have between 77 and 135 nodes. Given the relevance of the nodes, the prior conditional probability of a relevant observation is a beta distribution with parameters in Table 5.5.1b. We can compare the performance of the decision algorithms using BL and



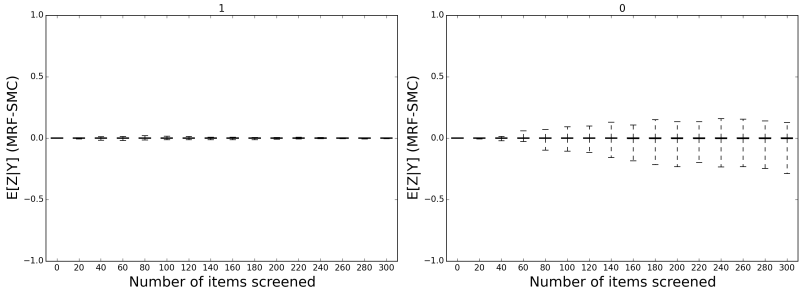
(a) BL



(b) SMC, $M = 1,000$



(c) SMC $M = 10,000$



(d) SMC $M = 100,000$

Figure 5.5.10: Tanzania network results: The distribution of differences, over 50 sets of random observations, given the relevance of the nodes, between inference in the Markov random field using prior conditional A and approximate updates using either the BL model or the SMC model with a range of maximum particles M.

SMC. Both priors are taken to be the same, and found using the method in Section 4.6.2 with $\mu = 0.25$ and $\delta = 0.8$. The greedy and Bayes-UCB decision algorithms are run for 50 repetitions with 200 observations.

Figure 5.5.12d shows the mean cumulative relevance over the 50 runs. For these networks, we can see the effect of sample impoverishment. As the number of maximum particles decreases, the number of relevant observations found also decreases. However, it is only when the maximum number of particles is reduced to $M = 100$ that the SMC model finds fewer items than the BL model. The BL model does not do much worse than the SMC model, even with this non-linear prior distribution, suggesting the BL model and SMC model with a small number of particles are able to capture enough of the relationship so that the decision algorithm can make sensible choices.

5.5.5 Enron Networks

In this section, we compare the performance of the SMC model and the BL model with non-linear prior conditional probabilities from networks simulated from the Enron Corpus. For the networks used in this section, there is no true underlying network.

In many cases, the BL model gives a particularly good approximation when the relationship is relatively linear between the latent variable and observations, see Section 4.7. We compare the BL model with the SMC model and the independence model introduced in Section 4.6, using a non-linear relationship (prior conditional A in Table 5.5.1b) for the Enron networks in Figures 5.5.14a and 5.5.13a which are also used in Section 4.7. For both networks, we find that after $2,000$ observations the BL model finds, on average, at least 50 more relevant observations than the SMC model and the independence model. Furthermore, we see that the SMC model screens roughly the same number of relevant items as the independence model after $2,000$ observations using the greedy heuristic, see Figures 5.5.13b and 5.5.14b. However, the SMC model does initially do better than the independence model. Considering the network structure is advantageous, particularly early on in the screening process. After a while, the independent model also finds the edges with the relevant items on and catches

up with the other methods

For this non-linear prior conditional distribution (prior conditional A in Table 5.5.1b), Section 5.5.1 suggest the BL model tends to give conservative estimates of the posterior values. The red nodes/edges in the Figures 5.5.13a and 5.5.14a show the nodes/edges associated with a high number of relevant items suggesting the Enron data networks only have a small amount of dependence between nodes in the network structure. Hence, the BL model giving conservative estimates to the influence of an observation on an edge has on other edges. The SMC model will give a better estimate of the relationship, which in this case may be too strong for the network structure resulting in a worse performance than the simpler BL model.

Alternative Prior Conditional Probability

For the Enron networks considered, the BL model tends to out perform the SMC model, even when the relationship between the latent variables and observations is non-linear. We consider an alternative probability distribution, where if at least one node is relevant, the mean of the conditional probability distribution is high. The prior conditional distribution (prior conditional B) is given in Table 5.5.1c. Instead of underestimating the updated values when the node is relevant, it might be expected that the BL model for prior conditional B poorly approximates the nodes with a relevance of 0. We run updates on the Enron networks with 449 and 669 nodes, and compare the mean cumulative number of relevant items for the SMC model and the BL model using the greedy and Bayes-UCB heuristics, see Figures 5.5.15 and 5.5.16. The SMC model finds more relevant items on average than the BL model. Both the BL model and the SMC models give good performance to begin with. However, after $2,000$ observations the independence model has caught up with the BL model for both networks with the greedy heuristic. There are enough observations that the independence model has found a number of good edges and sticks with them resulting in a rapid increase in the number of relevant item found towards the end.

The two decision policies using the SMC models are more robust to a change in the prior

conditional distribution; both prior conditional distributions give a similar number of relevant items. Whereas for the for the BL model, using prior conditional B gives far worse results than using prior conditional A.

5.6 Discussion

For non-linear relationships between the observations and the latent variables, we show that BL will give a poor approximation. We propose a SMC method a Monte Carlo approximation which better copes with non-linear relationships. A Monte Carlo sample is only taken to the subset of the random variables that are directly involved in the observations. BL methods are used to sample the binary random variables in the Monte Carlo sample, as well as provide posterior summaries for the remaining random variables not in the Monte Carlo approximation. We empirically show that for a small network with a non-linear relationship between the latent variables and the observations the SMC method gives a better approximation to the posterior mean and covariance than other simpler methods. Furthermore, we show that the BL and SMC models become are far more computationally efficient than the MRF model as the size of the network grows. When the SMC model is run on networks based on the Enron database, it does no better than the less computationally expensive constrained BL model, introduced in Chapter 4. Initially, both methods find more relevant items that the methods which ignore network structure.

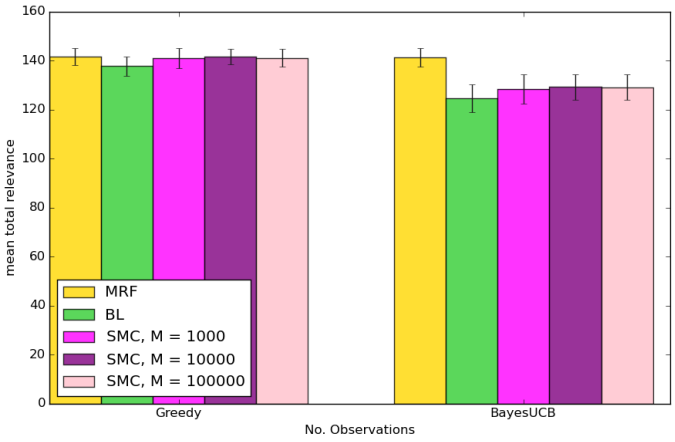
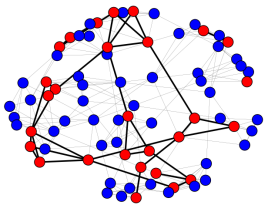
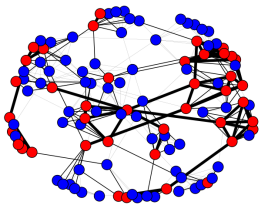


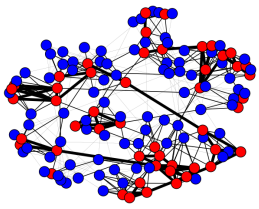
Figure 5.5.11: Tanzania Network: The mean cumulative number of relevant items screened ± 1 standard deviation using the three models.



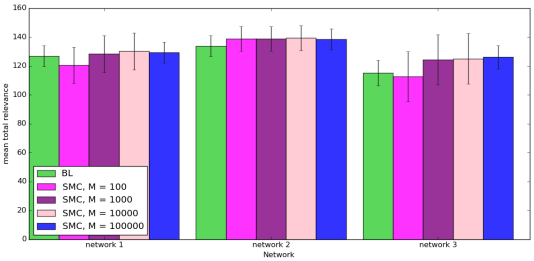
(a) Caveman Network 1



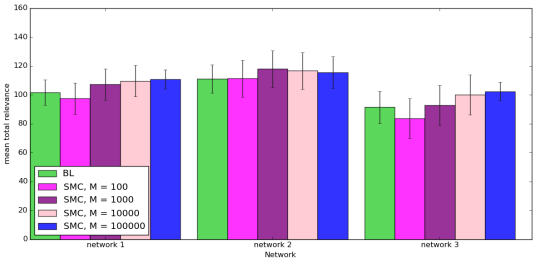
(b) Caveman Network 2



(c) Caveman Network 3

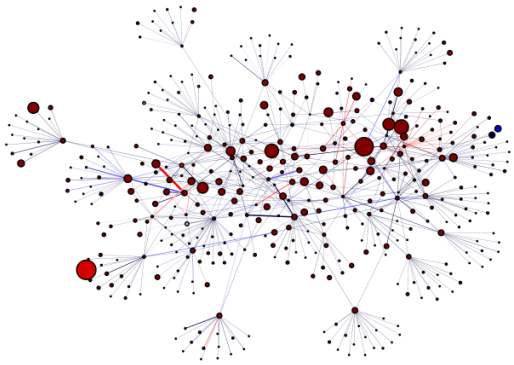


(d) Greedy

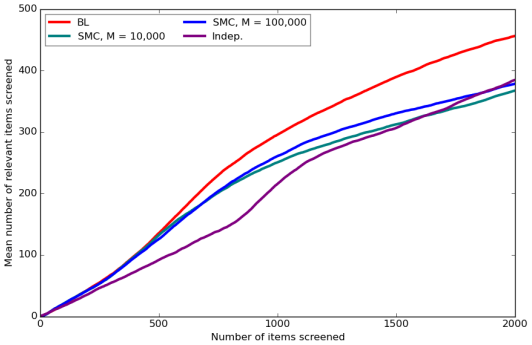


(e) Bayes-UCB

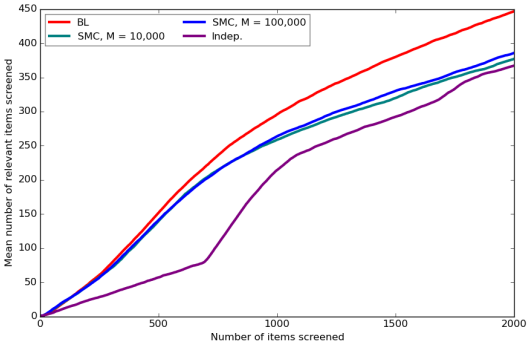
Figure 5.5.12: Caveman network results: The mean cumulative relevance ±1 standard deviation, over 50 runs of the greedy and Bayes-UCB algorithms, using prior conditional A and approximate updates using the BL model or the SMC model. The SMC model gives superior performance when the number of particles is set high enough.



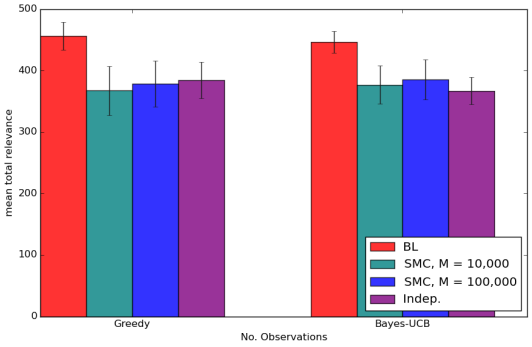
(a) Enron network 448 nodes



(b) Cumulative Relevance: Greedy

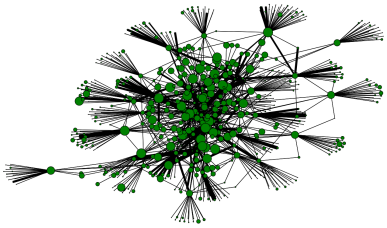


(c) Cumulative Relevance: Bayes-UCB

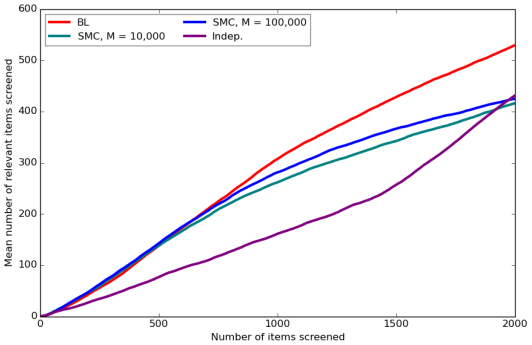


(d) Cumulative Relevance: boxplot

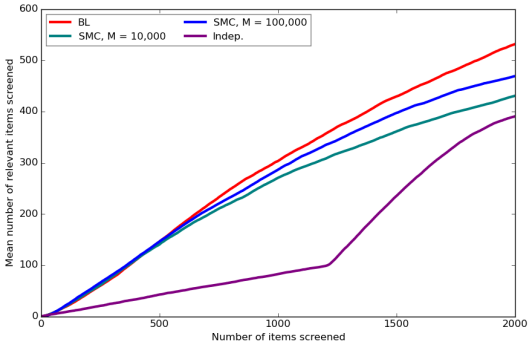
Figure 5.5.13: The mean cumulative relevance over 50 runs of the greedy and Bayes-UCB decision algorithms using prior conditional A on the networks shown in Figures 5.5.13a. The SMC algorithm gives a smaller cumulative relevance than the BL model.



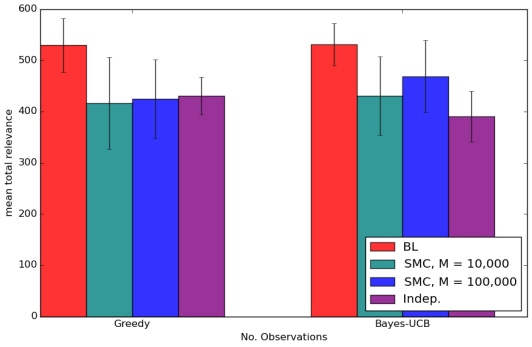
(a) Enron network 667 nodes



(b) Cumulative Relevance: Greedy

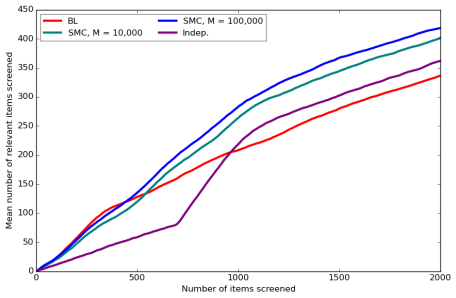


(c) Cumulative Relevance: Bayes-UCB

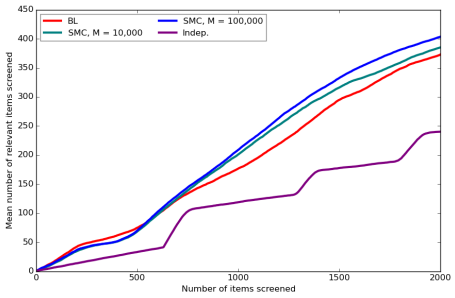


(d) Cumulative Relevance: boxplot

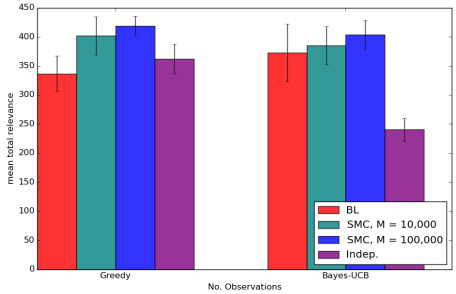
Figure 5.5.14: The mean cumulative relevance over 50 runs of the greedy decision algorithms using prior conditional A on the networks shown in Figures 5.5.14a. The SMC algorithm gives a smaller cumulative relevance than the BL model.



(a) Cumulative Relevance: Greedy

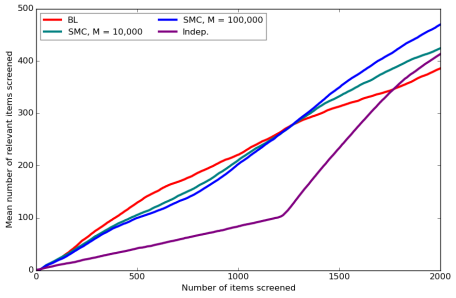


(b) Cumulative Relevance: Bayes-UCB

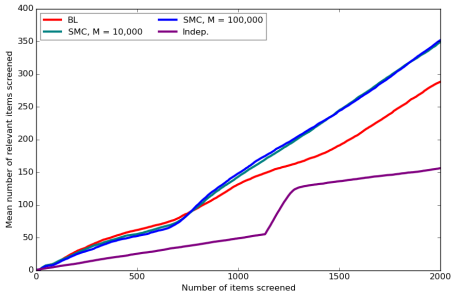


(c) Cumulative Relevance: boxplot

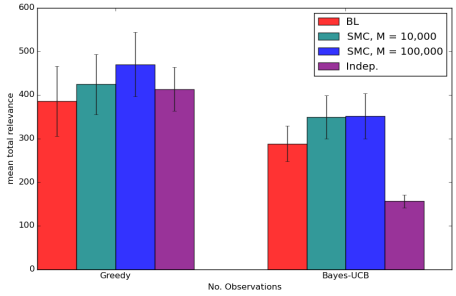
Figure 5.5.15: The mean cumulative relevance over 50 runs of the greedy and Bayes-UCB decision algorithms using prior conditional B on the networks shown in Figures 5.5.13a. The SMC algorithm gives a larger cumulative relevance than the BL model.



(a) Cumulative Relevance: Greedy



(b) Cumulative Relevance: Bayes-UCB



(c) Cumulative Relevance: boxplot

Figure 5.5.16: The mean cumulative relevance over 50 runs of the greedy and Bayes-UCB decision algorithms using prior conditional B on the networks shown in Figures 5.5.14a. The SMC algorithm gives a larger cumulative relevance than the BL model.

Chapter 6

Moments for Simulated Bernoulli Random Variables

6.1 Introduction

The Bayes Linear methods from Chapter 4 and Chapter 5 can be used to define a joint distribution for binary random variables with a given mean and covariance. This distribution is specified by ordering the random variables, $Z_1, \ldots, Z_n$, and uses Bayes Linear updates to calculate the distribution of Z_i given $Z_1, \ldots, Z_{i-1}$. A description of the method used to simulate the random variables is given in Section 5.3.1. In this chapter, we look in more detail at how the ordering of the random variables affects the resulting joint distribution.

Through a simple example with only three random variables, we see that the ordering of the random variables will affect the resulting joint probability distribution approximated using the BL methodology. For $n = 3$, if the mean and covariance of the distribution is fixed, there is one further degree of freedom in the distribution. One way of specifying this final degree of freedom is through the third centralised moment, or equivalently the probability $p(1, 1, 1)$. Knowing either of these will uniquely specify the full joint distribution. Assuming

the mean and covariance of the random variables, $\mathbf{Z} = (Z_1, Z_2, Z_3)$, are:

$$\mu = \begin{pmatrix} 0.4 \\ 0.5 \\ 0.6 \end{pmatrix} \ , \ \Sigma = \begin{pmatrix} 0.24 & 0.07 & 0.02 \\ 0.07 & 0.25 & 0.04 \\ 0.02 & 0.04 & 0.24 \end{pmatrix}. \tag{6.1.1}$$

The effect of ordering on the joint distribution can be seen by considering two possible orderings, (Z_1, Z_2, Z_3) and (Z_3, Z_2, Z_1), and comparing the value for $\hat{p}(1, 1, 1)$. For the ordering (Z_1, Z_2, Z_3), the Cholesky decomposition of covariance matrix is:

$$L = \begin{pmatrix} 0.489898 & 0.0 & 0.0 \\ 0.142887 & 0.479149 & 0.0 \\ 0.040825 & 0.071307 & 0.48296 \end{pmatrix}. \tag{6.1.2}$$

The approximate joint probability, found using BL with the ordering (Z_1, Z_2, Z_3) is:

$$\hat{p}(Z_1, Z_2, Z_3) = \hat{p}(Z_1)\hat{p}(Z_2 \mid Z_1)\hat{p}(Z_3 \mid Z_1, Z_2) \tag{6.1.3}$$

where the BL approximation to the probability $\hat{p}(Z_{k+1} \mid Z_{1:k})$ is given in (5.3.4). For the realisation $(Z_1 = 1, Z_2 = 1, Z_3 = 1)$, this ordering results in the BL approximation to the probability:

$$\begin{aligned} \hat{p}(Z_1 = 1, Z_2 = 1, Z_3 = 1) = \mu_1 \left(\mu_2 + L_{21}(L_{11})^{-1}(1 - \mu_1)\right) \big(\mu_3 + \big(L_{31}(L_{11})^{-1} \\ -L_{32}(L_{22})^{-1}L_{21}(L_{11})^{-1}\big)\left(Z_1 - \mu_1\right) + L_{32}(L_{22})^{-1}(Z_2 - \mu_2)\big)\,, \\ = 0.188559. \end{aligned} \tag{6.1.4}$$

Whereas, under the ordering (Z_3, Z_2, Z_1), the Cholesky decomposition will be:

$$A = \begin{pmatrix} 0.489898 & 0.0 & 0.0 \\ 0.081650 & 0.49329 & 0.0 \\ 0.040825 & 0.13515 & 0.46911 \end{pmatrix}, \tag{6.1.5}$$

Z_1	Z_2	Z_3	$\hat{p}(Z_1, Z_2, Z_3)$	$\hat{p}(Z_3, Z_2, Z_1)$	$\hat{p}(Z_1, Z_2, Z_3) - \hat{p}(Z_3, Z_2, Z_1)$
0	0	0	0.18144102	0.18230137	−0.00086035
1	0	0	0.05855898	0.05769863	0.00086035
0	1	0	0.07855898	0.07769863	0.00086035
1	1	0	0.08144102	0.08230137	−0.00086035
0	0	1	0.18855898	0.18769863	0.00086035
1	0	1	0.07144102	0.07230137	−0.00086035
0	1	1	0.15144102	0.15230137	−0.00086035
1	1	1	0.18855898	0.18769863	0.00086035

Table 6.1.1: Probability distributions of $\mathbf{Z} = (Z_1, Z_2, Z_3)$, approximated using BL with two different orderings but the same prior mean and covariance in (6.1.1). The probability distribution $\hat{p}(Z_1, Z_2, Z_3)$ is given by (6.1.3) whilst the probability distribution $\hat{p}(Z_3, Z_2, Z_1)$ is given by (6.1.6).

and the approximate BL probability will be:

$$\hat{p}(Z_3, Z_2, Z_1) = \hat{p}(Z_3)\hat{p}(Z_2 \mid Z_3)\hat{p}(Z_1 \mid Z_3, Z_2). \tag{6.1.6}$$

For the realisation $(Z_3 = 1, Z_2 = 1, Z_1 = 1)$, this ordering results in the BL approximate probability:

$$\begin{aligned}\hat{p}(Z_3 = 1, Z_2 = 1, Z_1 = 1) = \mu_3 \left(\mu_2 + A_{21}(A_{11})^{-1}(1 - \mu_3)\right) \big(\mu_3 + \big(A_{31}(A_{11})^{-1} \\ -A_{32}(A_{22})^{-1}A_{21}(A_{11})^{-1}\big)(Z_3 - \mu_3) + A_{32}(A_{22})^{-1}(Z_2 - \mu_2)\big) \\ = 0.187699. \end{aligned} \tag{6.1.7}$$

The different orderings give different probabilities for the same realisation. Table 6.1.1 shows the full probability distribution of the three random variables, approximated using the BL method with the two orderings and the difference between the two probabilities for each realisation. The difference in probabilities for each realisation is the same (up to sign) and will also equal to the difference in the third centralised moment. The third centralised moment for the ordering (Z_1, Z_2, Z_3) is 0.00055898 whilst for the ordering (Z_3, Z_2, Z_1) is −0.00030137. Hence, to look at the difference in the distributions, we focus on the effect ordering has on the third centralised moment.

Although the BL method results in errors in the joint distribution, we show in Section 6.2

that the mean and covariance of the resulting probability distribution will be the same as the mean and covariance used to simulate them as long as the prior conditions given in Section 5.3.1 hold. To investigate the aspects of the ordering which affect the distribution, we concentrate on the third moments of the distribution. In Section 6.3 we empirically show that the BL method will result in different third centralised moments depending on the ordering of the random variables. We then theoretically show some of the properties of the ordering that do/do not affect the 3rd centralised moment. In Section 6.4 we look at how much of an effect the ordering has on the third moment.

6.2 First and Second Moments

In Section 5.3.1 we give conditions on the prior mean and covariance that ensure the probability used to simulate the random variables are in the range $[0, 1]$. Lemma 6.2.1 states that the $\boldsymbol{\epsilon}$ simulated in Algorithm 7 has a zero mean and Lemma 6.2.2 that the covariance matrix of the $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$ is the identity matrix. The proofs are given in Appendix A.2.1.

Lemma 6.2.1. *For a prior mean, μ, and covariance, $\Sigma = LL^T$, which satisfy the conditions in Lemma 5.3.2, the random variable ϵ_i simulated from*

$$\epsilon_i = \begin{cases} (L_{ii})^{-1}(1 - \mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j) & \text{with probability } p \\ (L_{ii})^{-1}(-\mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j) & \text{with probability } 1-p \end{cases}, \tag{6.2.1}$$

where $p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j \in [0, 1]$, will have marginal expectation $E[\epsilon_i] = 0$.

Lemma 6.2.2. *For a prior mean, μ, and covariance, $\Sigma = LL^T$, which satisfy the conditions in Lemma 5.3.2, the random variable ϵ_i simulated from*

$$\epsilon_i = \begin{cases} (L_{ii})^{-1}(1 - \mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j) & \text{with probability } p \\ (L_{ii})^{-1}(-\mu_i - \sum_{j=1}^{i-1} L_{ij}\epsilon_j) & \text{with probability } 1-p \end{cases}, \tag{6.2.2}$$

where $p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j \in [0, 1]$, the covariance of $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$ is the identity matrix.

If the joint mean and covariance of the simulated random variables $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_n)$ are zero and the identity matrix respectively, we can show that the realisation of $\mathbf{Z}$ will have the marginal expectation:

$$E[\mathbf{Z}] = E[\mu + L\boldsymbol{\epsilon}] = \mu + LE[\boldsymbol{\epsilon}] = \mu, \tag{6.2.3}$$

and variance:

$$Var(\mathbf{Z}) = Var(\mu + L\boldsymbol{\epsilon}) = LVar(\boldsymbol{\epsilon}) L^T = LL^T = \Sigma. \tag{6.2.4}$$

Hence, the random variables simulated using Algorithm 7, will have the same mean and covariance used to simulate the random variables if the conditions in Section 5.3.2 hold.

6.3 Third Moment

In order to look at the effect of ordering on the joint distribution of the random variables, we concentrate on the third centralised moment. Firstly, we explore some of the effects ordering has through a simple example with $n = 5$ random variables, $\mathbf{Z} = (Z_1, Z_2, Z_3, Z_4, Z_5)$. The mean and covariance used to simulate the random variables are:

$$\mu = \begin{pmatrix} 0.4 \\ 0.5 \\ 0.5 \\ 0.6 \\ 0.5 \end{pmatrix}, \ \Sigma = \begin{pmatrix} 0.24 & 0.02 & 0.01 & 0.05 & 0.05 \\ 0.02 & 0.25 & 0.03 & 0.04 & 0.02 \\ 0.01 & 0.03 & 0.25 & 0.01 & 0.02 \\ 0.05 & 0.04 & 0.01 & 0.24 & 0.02 \\ 0.05 & 0.02 & 0.02 & 0.02 & 0.25 \end{pmatrix}. \tag{6.3.1}$$

Table 6.3.1 gives the third centralised moment, $\hat{E}[(Z_1 - \mu_1)(Z_2 - \mu_2)(Z_3 - \mu_3)]$, calculated from the joint distribution approximated using BL with different orderings of the random variables. From this table, we can see that the third centralised moment is affected by what is simulated before the first random variable considered. This can be seen by looking at the third centralised moments for the orderings $1, 2, 3, 4, 5$ and $5, 1, 2, 3, 4$ in Table 6.3.1,

Simulation Order	Third Centralised Moment
$(1,2,3,4,5)$	0.0001275168
$(1,2,4,3,5)$	0.0001275168
$(1,2,4,5,3)$	0.0001275168
$(2,1,3,4,5)$	0.0001275168
$(1,5,2,4,3)$	0.0001196806
$(1,5,2,3,4)$	0.0001196806
$(5,1,2,3,4)$	0.0001196806
$(4,5,1,2,3)$	0.0000628751

Table 6.3.1: The third centralised moment, $E[(Z_1-\mu_1)(Z_2-\mu_2)(Z_3-\mu_3)]$, for the random variables simulated with prior mean and covariance given in (6.3.1) using different orderings to calculate the joint probability distribution.

which are different. We can also see that the third centralised moment is affected by what is simulated between the first and second random variables considered, which can be seen by looking at the orderings $1,2,3,4,5$ and $1,5,2,3,4$ in Table 6.3.1.

From this table, we can also conjecture that after the first two random variables have been simulated, the third centralised moment will not be affected by the position of the third random variable in the ordering. This is demonstrated by considering the orderings $(1,2,3,4,5)$ and $(1,2,4,3,5)$ and the orderings $(1,5,2,4,3)$ and $(1,5,2,3,4)$ which give the same value for $E[(Z_1-\mu_1)(Z_2-\mu_2)(Z_3-\mu_3)]$. In Section 6.3.1, we show theoretically that this property holds.

6.3.1 Ordering of the Third Random Variable

The orderings $(1,2,3,4,5)$ and $(1,2,4,3,5)$ and the orderings $(1,5,2,4,3)$ and $(1,5,2,3,4)$ give the same value for $E[(Z_1-\mu_1)(Z_2-\mu_2)(Z_3-\mu_3)]$ in Table 6.3.1. This suggests that if the position of the first two random variables in the ordering are unchanged, the third centralised moment will not depend on how many random variables are simulated between the second and third random variables under consideration. In order to theoretically show this property holds, we firstly consider the effect of switching the final two random variables in the ordering on the Cholesky decomposition of the covariance matrix. Assume $\mathbf{Z}_{1:n}$ are simulated in numerical order, where L is the Cholesky decomposition of the covariance

matrix under this ordering. The covariance between Z_i and Z_j can be written in terms of L as:

$$\Sigma_{ij} = \sum_{k=1}^{j} L_{ik}L_{jk}.\ j < i \tag{6.3.2}$$

Using this, we can write the Cholesky decomposition, A, of the covariance matrix for the ordering $(Z_1, \ldots, Z_{n-2}, Z_n, Z_{n-1})$. The first $n-2$'s rows of A will be the same as the first $n-2$ rows of L. For the $n-1$th row, we get:

$$A_{n-1,i} = (L_{ii})^{-1}\left(\sum_{k=1}^{i} L_{nk}L_{ik} - \sum_{k=1}^{i-1} A_{n-1,k}A_{ik}\right) \tag{6.3.3}$$

which gives:

$$A_{n-1,i} = L_{ni} \text{ for } i < n-1. \tag{6.3.4}$$

Similarly, we can solve the equations for the nth row of A to give:

$$A_{n-1,i} = L_{ni} \text{ for } i < n-1. \tag{6.3.5}$$

Finally, the remaining three terms of A are not equivalent to just switching the rows of the Cholesky decomposition when the random variables are ordered numerically. Instead they are given by:

$$A_{n-1,n-1} = \sqrt{(L_{n,n-1}^2 + L_{nn}^2)} \tag{6.3.6}$$

$$A_{n,n-1} = \frac{L_{n-1,n-1}L_{n,n-1}}{\sqrt{(L_{n,n-1}^2 + L_{nn}^2)}} \tag{6.3.7}$$

$$A_{nn} = \sqrt{L_{n-1,n-1}^2 - \frac{L_{n-1,n-1}^2 L_{n,n-1}^2}{(L_{n,n-1}^2 + L_{nn}^2)}} \tag{6.3.8}$$

Hence, columns 1 to $n-2$ of A are the columns of L with the $n-1$th and nth row switched. However, the $n-1$th and nth columns do differ to give the Cholesky decomposition for the

ordering $(Z_1, \ldots, Z_{n-2}, Z_n, Z_{n-1})$:

$$A = \begin{pmatrix} L_{11} & 0 & \cdots & 0 & 0 & 0 \\ L_{21} & L_{22} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ L_{n-2,1} & L_{n-2,2} & \cdots & L_{n-2,n-2} & 0 & 0 \\ L_{n1} & L_{n2} & \cdots & L_{n,n-2} & A_{n-1,n-1} & 0 \\ L_{n-1,1} & L_{n-1,2} & \cdots & L_{n-1,n-2} & A_{n,n-1} & A_{nn} \end{pmatrix} \tag{6.3.9}$$

Using this property of the Cholesky decomposition, we can write the conditional expectation of the nth random variable for the two orderings as a function of the random variables lower down in the ordering and their conditional expectations. Lemma 6.3.1 gives the conditional expectation of $Z_n \mid Z_{1:n-1}$ under the ordering $(Z_1, \ldots, Z_{n-2}, Z_{n-1}, Z_n)$.

Lemma 6.3.1. *If L is the Cholesky decomposition of the covariance matrix for the random variables ordered numerically, the conditional expectation of $Z_n \mid Z_{1:n-1}$ found using the BL method is given by the recursive relationship:*

$$\begin{aligned} E[Z_n \mid Z_{1:n-1}] =& \mu_n - \sum_{j=1}^{n-1} L_{n,j}(L_{jj})^{-1}(E[Z_j \mid Z_{1:j-1}] - \mu_j) \\ &+ \sum_{j=1}^{n-1} L_{nj}(L_{jj})^{-1}(Z_j - \mu_j). \end{aligned} \tag{6.3.10}$$

Proof. The conditional expectation of Z_n given $\mathbf{Z}_{1:n-1}$ is:

$$E[Z_n \mid \mathbf{Z}_{1:n-1}] = \mu_n + \sum_{j=1}^{n-1} L_{nj}\epsilon_j \tag{6.3.11}$$

as $E[\epsilon_n \mid \epsilon_{1:n-1}] = 0$. Write ϵ_k in terms of $\epsilon_{1:k-1}$ and Z_k:

$$\epsilon_k = (L_{kk})^{-1}(Z_k - \mu_k - \sum_{i=1}^{k-1} L_{ki}\epsilon_i), \tag{6.3.12}$$

and substitute this into (6.3.11):

$$E[Z_n \mid \mathbf{Z}_{1:n-1}] = \mu_n + \sum_{j=1}^{n-1} L_{nj}(L_{jj})^{-1}(Z_j - \mu_j - \sum_{i=1}^{j-1} L_{ji}\epsilon_i) \tag{6.3.13}$$

$$= \mu_n + \sum_{j=1}^{n-1} L_{nj}(L_{jj})^{-1}(Z_j - \mu_j) - \sum_{j=1}^{n-1} L_{nj}(L_{jj})^{-1} \sum_{i=1}^{j-1} L_{ji}\epsilon_i \tag{6.3.14}$$

where $\sum_{i=1}^{j-1} L_{ji}\epsilon_i = E[Z_j \mid \mathbf{Z}_{1:j-1}] - \mu_j$. Substituting this into (6.3.14) gives the recursive relationship in (6.3.10). □

Alternatively, when the ordering is $(Z_1, \ldots, Z_{n-2}, Z_n, Z_{n-1})$, Lemma 6.3.2 can be used to write the conditional expectation of $Z_n \mid \mathbf{Z}_{1:n-2}$ in terms of the Cholesky decomposition for the random variables in numerical order.

Lemma 6.3.2. *If L is the Cholesky decomposition of the covariance matrix for the random variables ordered numerically, the full conditional of $Z_n \mid Z_{1:n-2}$ when the simulation ordering is $(Z_1, Z_2, \ldots, Z_{n-2}, Z_n)$ can be written using the recursive relationship:*

$$E[Z_n \mid Z_{1:n-2}] = \mu_n - \sum_{j=1}^{n-2} L_{n,j}(L_{jj})^{-1}(E[Z_j \mid Z_{1:j-1}] - \mu_j)$$

$$+ \sum_{i=1}^{n-2} L_{n,i}(L_{ii})^{-1}(Z_i - \mu_i). \tag{6.3.15}$$

Proof. Using the Cholesky decomposition in (6.3.9) for the ordering $(Z_1, Z_2, \ldots, Z_{n-2}, Z_n)$, the conditional expectation of Z_n given $\mathbf{Z}_{1:n-2}$ is:

$$E[Z_n \mid \mathbf{Z}_{1:n-2}] = \mu_n + \sum_{j=1}^{n-2} L_{nj}\epsilon_j \tag{6.3.16}$$

as $E[\epsilon_{n-1} \mid \epsilon_{1:n-2}] = 0$. Write the value of ϵ_j in terms of $\epsilon_{1:j-1}$ and Z_j:

$$E[Z_n \mid \mathbf{Z}_{1:n-2}] = \mu_n + \sum_{j=1}^{n-2} L_{nj}(L_{jj})^{-1}(Z_j - \mu_n - \sum_{i=1}^{j-1} L_{ji}\epsilon_i) \tag{6.3.17}$$

$$= \mu_n + \sum_{j=1}^{n-2} L_{nj}(L_{jj})^{-1}(Z_j - \mu_n) - \sum_{j=1}^{n-2} L_{nj}(L_{jj})^{-1} \sum_{i=1}^{j-1} L_{ji}\epsilon_i \tag{6.3.18}$$

where $\sum_{i=1}^{j-1} L_{ji}\epsilon_i = E[Z_j \mid \mathbf{Z}_{1:j-1}] - \mu_j$. Substituting this into (6.3.18) gives the recursive relationship in (6.3.15). □

We use these forms of the BL conditional expectation for the two orderings of the random variables to help show that the position of the third random variable in the ordering will not effect the third centralised moment. For three random variables Z_i, Z_j and Z_k, where $i < j < k$ in the ordering, Lemma 6.3.3 shows that if we know the position of the random variable $\mathbf{Z}_{1:j}$ and assume there are ordered numerically, then the position of Z_k in the ordering after this will not effect the BL approximation to $E[Z_k \mid Z_{1:j}]$.

Lemma 6.3.3. *For $i < j < k$, where the random variables $\mathbf{Z}_{1:j}$ are simulated numerically, the position of Z_k in the ordering will not affect the BL approximation to $E[Z_k \mid \mathbf{Z}_{1:j}]$ when the random variables are simulated using Algorithm 7.*

The proof is given in Appendix A.2.2. Using this property of the conditional expectation, we can show that the third centralised moment will also not be affected by the position of the third random variable, Z_k, in the ordering. This is given in Lemma 6.3.4.

Lemma 6.3.4. *For an ordering where $i < j < k$, the centralised third moment $E[(Z_i - \mu_i)(Z_j - \mu_j)(Z_k - \mu_k)]$ is not dependent on where the random variable Z_k is in the ordering after the position of Z_j.*

Proof. We suppose Z_k is the mth random variable in the ordering. Assume the random variables $\mathbf{Z}_{1:j}$ are ordered numerically, the third centralised moment is calculated using the conditional expectation of $E[Z_k \mid \mathbf{Z_{1:m}}]$ and then taking the expectation over $\mathbf{Z}_{1:m-1}$:

$$E[(Z_i - \mu_i)(Z_j - \mu_j)(Z_k - \mu_k)] = E[(Z_i - \mu_i)(Z_j - \mu_j)E[(Z_k - \mu_k) \mid \mathbf{Z}_{1:m}]] \tag{6.3.19}$$

Using Lemma 6.3.3 this conditional expectation is the same as:

$$E[(Z_i - \mu_i)(Z_j - \mu_j)(Z_k - \mu_k)] = E[(Z_i - \mu_i)(Z_j - \mu_j)E[(Z_k - \mu_k) \mid \mathbf{Z}_{1:l}]] \tag{6.3.20}$$

where $l > j$. Hence the third centralised moment is the same no matter the position of the random variable Z_k if it is after Z_i and Z_j in the ordering. □

6.4 How Much Does the Order Matter?

In Section 6.3 we showed that several properties of the ordering will affect the third centralised moment. As well as considering the properties that affect the ordering, we also look at how much of an effect the ordering will have on the value of the third centralised moment. In order to do this, we simulate a joint probability distribution by sampling probabilities for each possible realisation of the random variables from a Dirichlet distribution. The mean and covariance of the distribution can then be used to calculate the joint probability distribution using the BL method over all possible orderings of the random variables. We can then compare the differences in the third centralised moments for each of the orderings.

In the simplest case, where $n = 3$, there are six possible orderings of the random variables. In this case we simulate 500 distributions from a Dirichlet($\boldsymbol{\alpha}$) with $\alpha_i = 1,\ i = 1, \ldots, 8$. Figure 6.4.1 shows the distribution of differences in the third centralised moment for each combination of orderings. The distribution of the differences between the third central moment of the orderings range from -0.034 and 0.034 with the differences centred around zero. Apart from the orderings which give the same value for the third centralised moments, all orderings have similar distributions of differences. The difference, which is the maximum absolute difference over all order differences, is shown in Figure 6.4.2a.

For this set of distributions, we can look at the mean and covariance which give the two largest absolute differences between orderings. The mean and covariance:

$$\mu = \begin{pmatrix} 0.2870321 \\ 0.5459338 \\ 0.4814018 \end{pmatrix}, \ \Sigma = \begin{pmatrix} 0.2046447 & -0.11405505 & -0.11288769 \\ -0.1140551 & 0.24789009 & 0.01623224 \\ -0.1128877 & 0.01623224 & 0.24965411 \end{pmatrix} \tag{6.4.1}$$

give the largest absolute difference of 0.034 between four different combinations of orderings:

- $(Z_1, Z_3, Z_2) - (Z_2, Z_3, Z_1)$
- $(Z_1, Z_3, Z_2) - (Z_3, Z_2, Z_1)$

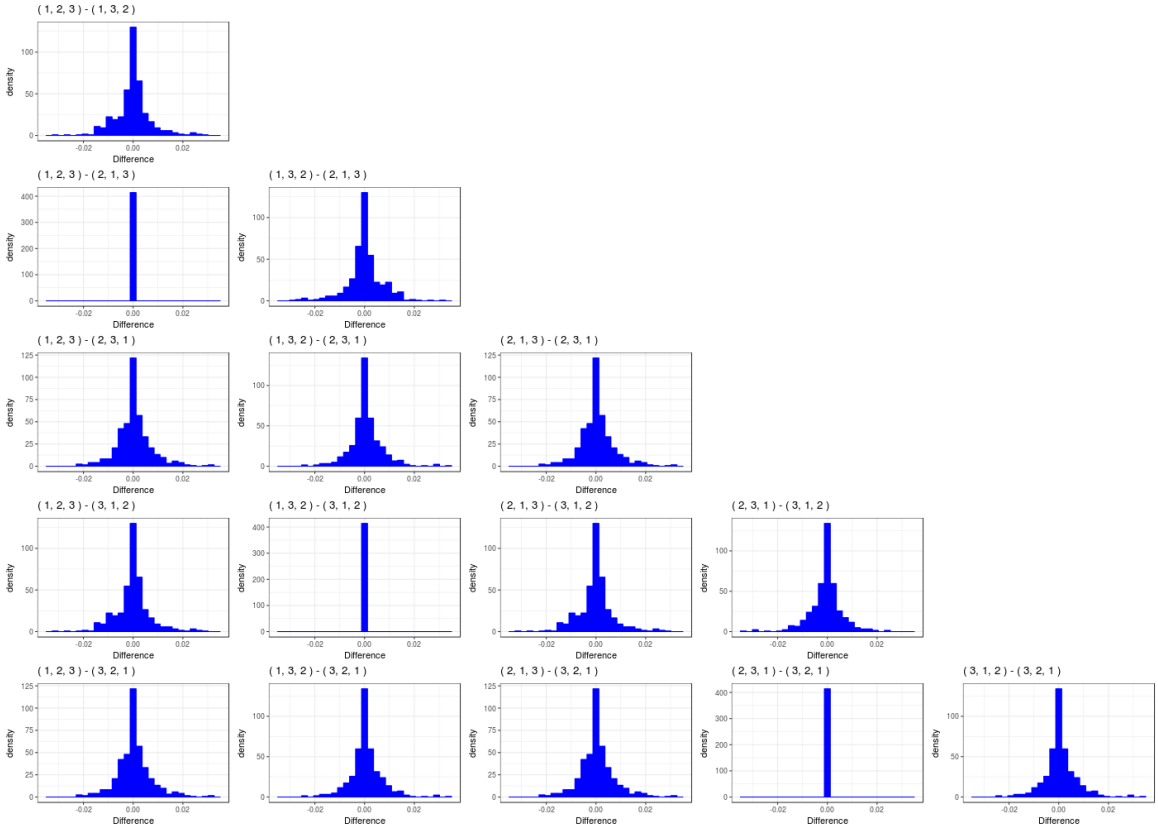


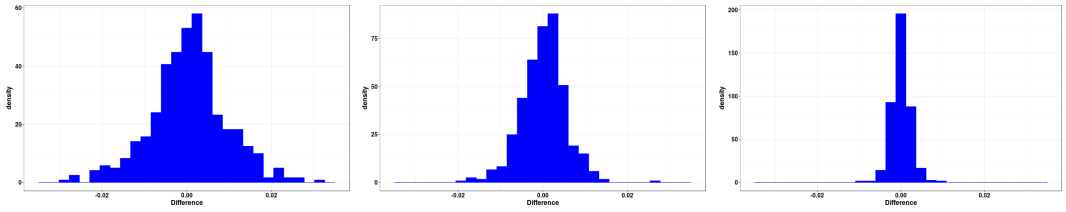
Figure 6.4.1: Difference in third centralised moments from the joint distributions simulated using BL methods under different orderings.

- $(Z_2, Z_3, Z_1) - (Z_3, Z_1, Z_2)$
- $(Z_3, Z_1, Z_2) - (Z_3, Z_2, Z_1)$

All of these orderings correspond to a change in the type of correlation between the first two random variables. For example, Z_1 and Z_3 are negatively correlated whilst Z_2 and Z_3 are positively correlated. The mean and covariance, which give the the second largest absolute differences of 0.031, also have a mixture of positively and negatively correlated random variables:

$$\mu = \begin{pmatrix} 0.3629383 \\ 0.6768653 \\ 0.4182529 \end{pmatrix}, \ \Sigma = \begin{pmatrix} 0.23121408 & 0.03507737 & -0.1332384 \\ 0.03507737 & 0.21871868 & 0.1234046 \\ -0.13323841 & 0.12340459 & 0.2433174 \end{pmatrix} \tag{6.4.2}$$

and the orderings which correspond to this difference are:



(a) $n = 3$ (b) $n = 4$ (c) $n = 5$

Figure 6.4.2: The maximum difference between the third centralised moments, $E[(Z_1 - \mu_1)(Z_2 - \mu_2)(Z_3 - \mu_3)]$, for two orderings of the n simulated random variables.

- $(Z_1, Z_2, Z_3) - (Z_2, Z_3, Z_1)$.
- $(Z_1, Z_2, Z_3) - (Z_3, Z_2, Z_1)$.
- $(Z_2, Z_1, Z_3) - (Z_2, Z_3, Z_1)$.
- $(Z_2, Z_1, Z_3) - (Z_3, Z_2, Z_1)$.

Once again, the orderings which give the largest differences are when the first two random variables in one ordering are positively correlated and negatively correlated in the second ordering.

For $n = 3$, the third centralised moment, along with the mean and covariance, will uniquely define the joint distribution. For $n > 3$, we can still look at the differences in the third centralised moments for the BL distributions under the different orderings. The mean and covariance are calculated from 500 probability distributions sampled from a Dirichlet($\boldsymbol{\alpha}$) distribution with $\alpha_i = 1$, $i = 1, \ldots, 2^n$. Figure 6.4.2 shows a distribution of differences, which is the maximum absolute difference for each simulated probability distributions, for between $n = 3$ to $n = 5$ random variables. As the number of random variables increases, the maximum differences in the third centralised moment decrease.

6.5 Discussion

The distribution found using BL methods is specified by ordering the random variables, $Z_1, \ldots, Z_n$, and uses BL updates to calculate the distribution of Z_i given $Z_1, \ldots, Z_{i-1}$. We

have shown that the ordering of the random variables affects the resulting joint distribution. However, under certain constraints on the mean and covariance, the random variables will have the same mean and covariance that is used to simulate them. We investigate, both theoretically and empirically, some of the aspects of the ordering which do/do not affect the joint distribution by looking at the third moments. Given the position of the first two random variables in the ordering, the position of the final random variable will not matter, but the third moment will be affected by what is simulated before and in between the first and second. The ordering of the random variables often results in differences which are very small. The largest differences in the third central moment for the same distribution occur when there is a mixture of both positive and negative correlations between the random variables. The ordering tends to affect the third centralised moment less as the number of random variables in the distribution increases.

Chapter 7

Conclusions

This section will summarise the contributions of this thesis, discuss what motivated the work and outline possible future work that leads from it. The work develops sequential inference methods which can be applied to the problem of network based search. The problem of network based sequential searches can be seen as having three related steps: Defining an appropriate joint prior distribution for the network; updating the joint distribution as observations are made; and deciding which items to screen next, given the current joint distribution. This thesis focuses on approximate inference methods for updating the joint distribution with the methodology influenced by the other steps.

Dimitrov et al. (2016) shows that using the information about the network structure, together with a modelling assumption that relevant items and participants are likely to cluster together, can greatly increase the rate of finding relevant items. However their approach is computationally expensive and thus limited in applicability to small networks. For a large network, defining a joint binary distribution over all nodes in the network is in itself a difficult problem. Instead, we consider the case when there is a limited prior specification. In Chapter 4, we develop a method based on Bayes Linear methodology which works directly with prior means and covariances rather than the full probability distributions. Furthermore, it provides a natural approach to modelling such data as the method provides the output posterior summaries that are most relevant to heuristic policies for choosing future

items.

Bayes linear methodology makes no assumptions about the underlying distribution so the posterior mean may not be in the range $[0, 1]$. We introduce a new optimisation problem for Bernoulli random variables, called constrained Bayes Linear, in Chapter 4. The optimization problem has additional constraints on the posterior summaries incorporated into the Bayes Linear optimisation problem. These additional constraints ensure the updated expectations are in the range $[0, 1]$; a desirable property for Bernoulli random variables. We show that, like the unconstrained Bayes linear methodology, the resulting optimisation problem has an analytical solution. The constrained Bayes linear method can be applied to the network based search problem. For small networks, we show the updated values using constrained Bayes linear are close to the values found using exact inference methods and the performance of the decision problem is not greatly affected by the inference method. For both simulated data and data from the Enron corpus the BL model gives comparable performance to the methods of Dimitrov et al. (2016), and for networks where this method is infeasible Bayes linear gives superior performance to methods that ignore the network structure.

Bayes linear methodology relies on a linear approximation between the random variables and the observations. Chapter 5 shows that for highly non-linear relationships, this gives a poor approximation. To overcome this problem, we propose a novel sequential Monte Carlo method for sequential inference on the network. Rather than take a Monte Carlo approximation to the entire network, we only include a Monte Carlo sample of a subset of these random variables – those directly involved with observations. This deals with the non-linear relationship whilst minimising the computational cost and problems like sample impoverishment.

The success of the Bayes linear method in Chapter 4 provides the motivation for using Bayes linear to sample the binary random variables from the prior mean and covariance given in Chapter 5. Bayes linear is also used to provide posterior summaries for the remaining random variables not in the Monte Carlo approximation. We give a method for simulating the random variables based directly on the Cholesky decomposition of the covariance matrix.

Constraints on the prior mean and Cholesky decomposition of the covariance matrix are given so that the mean and covariance of the simulated random variables match those used to simulate them.

We empirically show that for a small network with a non-linear relationship between the latent variables and the observations the sequential Monte Carlo method gives a better approximation to the posterior mean and covariance than other methods (including constrained Bayes linear and simpler Monte Carlo methods) when compared with exact inference in the binary Markov random field. For simulated networks, we show that even with a small number of maximum particles the sequential Monte Carlo method gives similar or better performance than the Bayes linear model in the decision problem. However, when the sequential Monte Carlo model is run on networks based on the Enron database, the sequential Monte Carlo model does no better than the less computationally expensive constrained Bayes linear model, introduced in Chapter 4. Therefore, even though the sequential Monte Carlo model better approximates the relationship between the observations, the simpler approximation is good enough to help guide the decisions. Initially, both methods find more relevant items that the methods which ignore network structure.

In Chapter 6 we explore the properties of the Bayes linear methods in Chapters 4 and 5 used to define the joint distribution of the binary random variables with a given mean and covariance. We show that under certain constraints of the mean and covariance, the random variables will have the same mean and covariance that is used to simulate them. The distribution found using Bayes linear methods is specified by ordering the random variables, $Z_1, \ldots, Z_n$, and using Bayes linear updates to calculate the distribution of Z_i given $Z_1, \ldots, Z_{i-1}$. We show in Chapter 6 that the ordering of the random variables affects the resulting joint distribution. Furthermore we investigate, both theoretically and empirically, some of the aspects of the ordering which do/do not affect the joint distribution by looking at the third moments. Given the position of the first two random variables in the ordering, the position of the final random variable will not matter, but the third moment will be effected by what is simulated before and in between the first and second. Finally, we considered how much of an effect the ordering of the random variables has on the BL approximation to the

third centralised moment.

7.1 Possible Future Directions

Throughout the thesis, we have described the Bayes linear approach for one specific model but it can easily be applied more widely. For example, it is straightforward to allow for differences between participants' nodes, such as those due to covariate information, by allowing for different mean probabilities of relevance for different nodes. Similarly, we can generalise the dependence structure assumed by the Bayes linear method, for example allowing nodes that communicate more frequently to be more strongly linked, by altering how the prior variance is specified. Furthermore, the Bayes linear approach could be applied to models which assumed a different distribution for the probability of relevance of an item on an edge, given the relevance of the participants it is between. This model could include covariate information about the item, or be different depending on which of the participants is the sender and which is the receiver.

Several assumptions were made about the model, including the assumption that items are only made between two people. In order for the method to be more applicable to some of the problems described in the introduction, such as legal precedent searches, this assumption would need to be looked at, as many legal cases are likely to involve more than two people. To some extent this is also true of email databases with group emails. This could be solved, for example, by updating all nodes involved with an item on one of the edges and altering the decision algorithm to consider the additional information, if any, gained from screening this type of observation.

More generally, further research could look at how the multi-armed bandit problems on networks could be used to better influence the decisions made. Differences between this problem and a classic multi-armed bandit problem include correlations between arms (edges in the network), a short time horizon and a varying number of items available on each arm. Multi-armed bandit problems have been proposed for the problem of learning the most

influential people within a social network. Many of which involve only sharing information with neighbouring nodes in the the network. Although the methods discussed in this thesis extend the size of networks far beyond that allowed by exact inference, the computational cost is still limited by calculating the Cholesky decomposition of the covariance matrix. By using local updates instead of global updates similar to those multi-armed bandit methods on social networks, the size of network could be extended even further.

Appendix A

Appendix

A.1 Chapter 4

A.1.1 Proofs for Chapter 4

Proof (Proof of Lemma 4.3.1)**.** *Consider two optimisation problems:* (4.3.8) *without its constraints, which is equivalent to the optimisation problem* (4.3.1)*, and* (4.3.8) *with the constraints. Both problems are convex minimisation problems with the same objective function, and if we assume that Var*(Z) *is positive definite, they are strongly convex. Suppose* (4.3.8) *has an optimal solution* h^**, for which none of the constraints are tight. The same solution is the optimal for* (4.3.1) *because the objective function is strongly convex. That is, you cannot move in any direction* d*, to a new solution for* (4.3.1)*,* $h^* + \epsilon d$*, and decrease the objective. This shows the contrapositive of the statement: If* (4.3.8) *has an optimal solution without tight constraints,* (4.3.8) *and* (4.3.1) *have the same optimal solution. If they do not have the same solution, then* (4.3.8) *has to have a tight constraint.* □

Proof (Proof of Lemma 4.3.2)**.** *The optimisation problem in* (4.3.10)*,*

$$\begin{aligned} \underset{\mathbf{h}^k}{\textit{minimise}} \quad & E\left[\left(Z_k - h_0^k - \sum_{i=1}^{n} h_i^k Y_i\right)^2\right], \\ \textit{subject to} \quad & h_0^k + \sum_{i=1}^{n} h_i^k y_i = c, \end{aligned}$$

can be solved optimally using the method of Lagrange multipliers to give an analytical solution. Let:

$$\Lambda(\mathbf{h}^k, \lambda) = E\left[\left(Z_k - h_0^k - (\mathbf{h}_{1:n}^k)^T \mathbf{Y}\right)^2\right] + \lambda(h_0^k + (\mathbf{h}_{1:n}^k)^T \mathbf{y} - c), \tag{A.1.1}$$

where $\mathbf{h}_{1:n}^k = (h_1^k, \ldots, h_n^k)^T$*. The optimisation problem is equivalent to solving:*

$$\underset{\mathbf{h}^k, \lambda}{\textit{minimise}}\ \Lambda(\mathbf{h}^k, \lambda). \tag{A.1.2}$$

Differentiating A.1.1 with respect to h_0^k*,* $\mathbf{h}_{1:n}^k$ *and* λ *gives:*

$$\frac{\partial \Lambda(h_0^k, \mathbf{h}_{1:n}^k, \lambda)}{\partial h_0^k} = 2h_0^k - 2E[Z_k] + 2(\mathbf{h}_{1:n}^k)^T E[\mathbf{Y}] + \lambda \tag{A.1.3}$$

$$\frac{\partial \Lambda(h_0^k, \mathbf{h}_{1:n}^k, \lambda)}{\partial \lambda} = h_0^k + (\mathbf{h}_{1:n}^k)^T \mathbf{y} - c \tag{A.1.4}$$

$$\bigtriangledown_{\mathbf{h}_{1:n}^k} \Lambda(h_0^k, \mathbf{h}_{1:n}^k, \lambda) = -2E[Z_k \mathbf{Y}] + 2h_0^k E[\mathbf{Y}]^T + 2(\mathbf{h}_{1:n}^k)^T E[\mathbf{Y}\mathbf{Y}] + \lambda \mathbf{y}^T \tag{A.1.5}$$

The optimal solution is found by the partial derivatives (A.1.3) - (A.1.5) *equal to zero,*

$$0 = 2h_0^k - 2E[Z_k] + (\mathbf{h}_{1:n}^k)^T E[\mathbf{Y}] + \lambda, \tag{A.1.6}$$

$$0 = h_0^k + (\mathbf{h}_{1:n}^k)^T \mathbf{y} - c, \tag{A.1.7}$$

$$0 = -2E[Z_k \mathbf{Y}] + 2h_0^k E[\mathbf{Y}]^T + 2(\mathbf{h}_{1:n}^k)^T E[\mathbf{Y}\mathbf{Y}] + \lambda \mathbf{y}^T, \tag{A.1.8}$$

and solving the set of simultaneous equations. Rearranging (A.1.6) *to give* h_0^k *in terms of* λ

and $\mathbf{h}^k_{1:n}$ *and substituting into equations* (A.1.7) *and* (A.1.8) *gives:*

$$0 = E[Z_k] + (\mathbf{h}^k_{1:n})^T(\mathbf{y} - E[\mathbf{Y}]) - \frac{\lambda}{2} - c \tag{A.1.9}$$

$$\begin{aligned} 0 &= -2E[Z_k\mathbf{Y}] + 2\left(E[Z_k] - (\mathbf{h}^k_{1:n})^T E[\mathbf{Y}] - \frac{\lambda}{2}\right) E[\mathbf{Y}]^T + 2(\mathbf{h}^k_{1:n})^T E[\mathbf{Y}\mathbf{Y}] + \lambda\mathbf{y}^T \\ &= -2Cov(Z_k, \mathbf{Y}) + 2(\mathbf{h}^k_{1:n})^T Var(\mathbf{Y}) + \lambda(\mathbf{y} - E[\mathbf{Y}])^T \end{aligned} \tag{A.1.10}$$

Rearranging (A.1.9)*, we get:*

$$\lambda = 2E[Z_k] + 2(\mathbf{h}^k_{1:n})^T(\mathbf{y} - E[\mathbf{Y}]) - 2c, \tag{A.1.11}$$

and substituting into (A.1.10)*,*

$$0 = -2Cov(Z_k, \mathbf{Y}) + 2(\mathbf{h}^k_{1:n})^T Var(\mathbf{Y}) + \Big(2E[Z_k] + 2(\mathbf{h}^k_{1:n})^T(\mathbf{y} - E[\mathbf{Y}]) - 2c\Big)(\mathbf{y} - E[\mathbf{Y}])^T$$

Finally, rearranging we get an expression for the optimal value of $(\mathbf{h}^k_{1:n})^T$*, which satisfies* (4.3.10)

$$\big(Cov(Z_k, \mathbf{Y}) + (c - E[Z_k])(\mathbf{y} - E[\mathbf{Y}])^T\big)\big(Var(\mathbf{Y}) + (\mathbf{y} - E[\mathbf{Y}])(\mathbf{y} - E[\mathbf{Y}])^T\big)^{-1}. \tag{A.1.12}$$

Substituting into equation A.1.7 gives the optimal value for h^k_0*:*

$$h^k_0 = c - (\mathbf{h}^k_{1:n})^T\mathbf{y}. \tag{A.1.13}$$

□

Proof (Proof of Lemma 4.4.1)**.** *The expected value of* $\mathbf{Y}$ *can be calculated from:*

$$E[Y_{uv}] = E[E[Y_{uv} \mid P_{uv}]] \tag{A.1.14}$$

$$= n_{uv}E[P_{uv}]. \tag{A.1.15}$$

where n_{uv} *is the number of items observed on edge* (u, v) *to date. Similarly, the analytical*

formula for $E[Z_k Y_{uv}]$ *is:*

$$\begin{aligned} E[Z_k Y_{uv}] &= E[Z_k E[Y_{uv} \mid P_{uv}]], \\ &= n_{uv} E[Z_k P_{uv}]. \end{aligned} \tag{A.1.16}$$

The analytical formula for $E[\mathbf{YY}]$ *is defined in terms of diagonal and off diagonal terms of the matrix where:*

$$\begin{aligned} E\left[Y_{uv}^2\right] &= E\left[E\left[Y_{uv}^2 \mid P_{uv}\right]\right], \\ &= E\left[Var(Y_{uv} \mid P_{uv}) + E[Y_{uv} \mid P_{uv}]^2\right], \\ &= E\left[n_{uv}(n_{uv}-1)P_{uv}^2 + n_{uv}P_{uv}\right], \\ &= n_{uv}(n_{uv}-1)E\left[Var(P_{uv} \mid Z_u, Z_v) + E[P_{uv} \mid Z_u, Z_v]^2\right] + n_{uv}E[P_{uv}], \end{aligned} \tag{A.1.17}$$

and

$$\begin{aligned} E[Y_{uv}Y_{ij}] &= E[E[Y_{uv}Y_{ij} \mid P_{uv}, P_{ij}]], \\ &= E[E[Y_{uv} \mid P_{uv}]\, E[Y_{ij} \mid P_{ij}]], \\ &= n_{uv}n_{ij}E[P_{uv}P_{ij}]. \end{aligned} \tag{A.1.18}$$

□

A.1.2 Maximum Entropy Method for Approximating Joint Distribution

Several of the solutions in Lemma 4.4.1 require the joint distribution over several participants' relevance values and probabilities. However, we only have a prior mean and covariance of $\mathbf{Z}$ and the joint distribution for more than two binary $\mathbf{Z}$ values is not uniquely defined by their expectation and covariance. An alternative method to using BL updates to calculate this joint distribution is to calculate the maximum entropy distribution. This can be applied in the cases where the mean and covariance of the BL method are not within the bounds described in Chapter 5 as an alternative to using the constrained Bayes linear.

The maximum entropy joint binary distribution for the required $\mathbf{Z}$, which most closely agrees with their expectation and covariance. The maximum entropy distribution is the probability distribution which best represents the current state of knowledge. If X is a discrete random variable, the maximum entropy distribution given by:

$$p(X_k = x_k) = p_k \ , \ k = 1, 2, \ldots, n, \tag{A.1.19}$$

is found by maximising the entropy $(-\sum_{k=1}^{n} p_k \log p_k)$ subject to any constraints on the probability distribution.

For this application, the maximum entropy distribution can be used to approximate the joint binary distribution of up to four $\mathbf{Z}$ values. It is found by solving the convex optimisation problem:

$$\begin{aligned} \text{maximise} \quad & -\sum_{i=1}^{n} p_i \log p_i, \\ \text{subject to} \quad & 0 \leq p_i \leq 1 \ , \ i, j = 1, \ldots, n \\ & 1 = \sum_{i=1}^{n} p_i, \\ & E[Z_i] = p_i \ , \ i, j = 1, \ldots, n, \\ & Cov(Z_i, Z_j) = (p_i - E[Z_i])(p_j - E[Z_j]) \ , \ i, j = 1, \ldots, n. \end{aligned} \tag{A.1.20}$$

If the optimisation problem is over defined, a least squares approximation is used. From this and the conditional beta distributions, we can calculate the required expectations. We describe here the calculations for $E[P_{uv}P_{ij}]$. Similar calculations are use for $E\left[Z_k P_{uv}\right]$, $E\left[\text{Var}(P_{uv} \mid Z_u, Z_v)\right]$ and $E\left[E\left[P_{uv} \mid Z_u, Z_v\right]^2\right]$, and these are given in Appendix A.1.3. The values only have to be calculated once, prior to the screening process.

To calculate $E[P_{uv}P_{ij}]$, we would calculate the maximum entropy distribution of Z_u, Z_v, Z_i, Z_j using equation A.1.20, say $\tilde{p}(Z_u, Z_v, Z_i, Z_j)$. This can be done using standard convex optimisation packages. From this approximate joint distribution, the approximate value of

$E[P_{uv}P_{ij}]$ is:

$$\begin{aligned} E[P_{uv}P_{ij}] &= E\left[E\left[P_{uv}P_{ij}|Z_u, Z_v, Z_i, Z_j\right]\right] \\ &= \sum_{z_u,z_v,z_i,z_j \in \{0,1\}} \tilde{p}(z_u, z_v, z_i, z_j) E[P_{uv}|z_u, z_v] E[P_{ij}|z_i, z_j] \end{aligned}$$

where $E\left[P_{uv}P_{ij}|Z_u, Z_v, Z_i, Z_j\right] = E[P_{uv}|z_u, z_v]E[P_{ij}|z_i, z_j]$ by conditional independence.

A.1.3 Calculations for Analytical Equations Used in Lemma 4.4.1

Lemma 4.4.1 requires the values $E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v)\right]$, $E\left[E\left[P_{uv} \mid Z_u, Z_v\right]^2\right]$ and $E\left[Z_k P_{uv}\right]$. In order to calculate $E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v)\right]$, we require the joint distribution over Z_u, Z_v. For two random variables these can be calculated exactly from their prior expectation and covariance, to give $p(Z_u, Z_v)$, so:

$$E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v)\right] = \sum_{Z_u,Z_v \in \{0,1\}} p(z_u, z_v)\mathrm{Var}(P_{uv}|z_u, z_v). \tag{A.1.21}$$

Similarly, calculating $E\left[E\left[P_{uv} \mid Z_u, Z_v\right]^2\right]$ we can get $p(Z_u, Z_v)$ from the prior expectation and covariance. From this we get:

$$E\left[E[P_{uv} \mid Z_u, Z_v]^2\right] = \sum_{Z_u,Z_v \in \{0,1\}} p(z_u, z_v) E[P_{uv}|z_u, z_v]^2. \tag{A.1.22}$$

Calculating $E\left[Z_k P_{uv}\right]$ can require calculating the joint distribution over up to three Z. This is estimated using the maximum entropy distribution as $\tilde{p}(Z_k, Z_u, Z_v)$, and using this, we get:

$$E\left[Z_k P_{uv}\right] = \sum_{Z_u,Z_v,Z_k \in \{0,1\}} \tilde{p}(z_k, z_u, z_v) z_k E[P_{uv}|z_u, z_v]. \tag{A.1.23}$$

A.1.4 Calculations for Posterior Approximations

In Section 4.4.2 we described how to calculate the the posterior value for $\tilde{E}[P_{uv}P_{ij} \mid \mathbf{y}]$. Here we describe the similar calculations for $E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v, \mathbf{y})\right]$, $E\left[E\left[P_{uv} \mid Z_u, Z_v, \mathbf{y}\right]^2\right]$ and $E\left[Z_k P_{uv} \mid \mathbf{y}\right]$. In order to calculate $E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v, \mathbf{y})\right]$, we require the approximate posterior joint distribution over $Z_u, Z_v \mid \mathbf{y}$. The method for calculating this is given in Section 4.4.2. From this:

$$E\left[\mathrm{Var}(P_{uv} \mid Z_u, Z_v, \mathbf{y_{uv}})\right] = \sum_{Z_u, Z_v \in \{0,1\}} p(z_u, z_v \mid \mathbf{y}) \mathrm{Var}(P_{uv} | z_u, z_v, \mathbf{y_{uv}}). \tag{A.1.24}$$

Similarly, calculating $E\left[E\left[P_{uv} \mid Z_u, Z_v, \mathbf{y_{uv}}\right]^2\right]$ we can get $\tilde{p}(Z_u, Z_v \mid \mathbf{y})$ from the posterior expectation and covariance. From this we get:

$$E\left[E[P_{uv} \mid Z_u, Z_v, \mathbf{y}]^2\right] = \sum_{Z_u, Z_v \in \{0,1\}} p(z_u, z_v \mid \mathbf{y}) E[P_{uv} | z_u, z_v, \mathbf{y_{uv}}]^2. \tag{A.1.25}$$

Calculating $E\left[Z_k P_{uv} \mid \mathbf{y}\right]$ can require calculating the joint distribution over up to three Z values. This is estimated using BL updates as $\tilde{p}(Z_k, Z_u, Z_v \mid \mathbf{y})$, and using this, we get:

$$E\left[Z_k P_{uv} \mid \mathbf{y}\right] = \sum_{Z_u, Z_v, Z_k \in \{0,1\}} \tilde{p}(z_k, z_u, z_v \mid \mathbf{y}) z_k E[P_{uv} | z_u, z_v, \mathbf{y_{uv}}]. \tag{A.1.26}$$

A.2 Chapter 5

A.2.1 Proofs for Simulating Using the Cholesky Decomposition of Covariance Matrix

Proof (Proof of Lemma 6.2.1)**.** Letting $p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j$, we can rewrite (6.2.1) as:

$$\epsilon_i = \begin{cases} (L_{ii})^{-1}(1-p) & \text{with probability } p \\ -(L_{ii})^{-1} p & \text{with probability } 1-p \end{cases}, \tag{A.2.1}$$

and the conditional expectation $E[\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}]$ is calculated from A.2.1 as:

$$\begin{aligned} E[\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}] &= (L_{ii})^{-1}\left((1-p)p - p(1-p)\right) \\ &= 0. \end{aligned} \tag{A.2.2}$$

The marginal expectation is found by taking the expectation of $E[\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}]$ with respect to $\epsilon_1,\ldots,\epsilon_{i-1}$:

$$E[\epsilon_i] = E\left[E[\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}]\right] = E[0] = 0. \tag{A.2.3}$$

□

Proof (Proof for Lemma 6.2.2)**.** The covariance between ϵ_i and ϵ_j, for $j < i$, is:

$$Cov(\epsilon_i,\epsilon_j) = E[\epsilon_i\epsilon_j] - E[\epsilon_i]E[\epsilon_j]. \tag{A.2.4}$$

where $E[\epsilon_i] = 0$ and $E[\epsilon_j] = 0$ and $E[\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}] = 0$ so:

$$\begin{aligned} Cov(\epsilon_i,\epsilon_j) &= E[\epsilon_i\epsilon_j] \\ &= E[E[\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}]\epsilon_j] \\ &= 0. \end{aligned} \tag{A.2.5}$$

The marginal variance, $Var(\epsilon_i)$ can be found from $\epsilon_i \mid \epsilon_1,\ldots,\epsilon_{i-1}$, using:

$$\begin{aligned} Var(\epsilon_i) &= E\left[Var\left(\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}\right)\right] + Var\left(E\left[\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}\right]\right), \\ &= E\left[Var\left(\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}\right)\right]. \end{aligned} \tag{A.2.6}$$

The variance of $\epsilon_i|\epsilon_1,\ldots,\epsilon_{i-1}$, where $p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j$ is:

$$\begin{aligned} Var(\epsilon_i) &= E\left[(L_{ii})^{-2}(1-p)^2 p + (L_{ii})^{-2}p^2(1-p)\right], \\ &= (L_{ii})^{-2}E\left[p(1-p)\right]. \end{aligned} \tag{A.2.7}$$

Substituting in the value of $p = \mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j$ we get:

$$\begin{aligned} Var(\epsilon_i) &= (L_{ii})^{-2} E\left[\mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j - \left(\mu_i + \sum_{j=1}^{i-1} L_{ij}\epsilon_j\right)^2\right], \\ &= (L_{ii})^{-2}\left(\mu_i - \mu_i^2 + (1-2\mu_i)\sum_{j=1}^{i-1} L_{ij}E\left[\epsilon_j\right] - E\left[\sum_{j=1}^{i-1}\sum_{k=1}^{i-1} L_{ij}L_{ik}\epsilon_j\epsilon_k\right]\right). \quad \text{(A.2.8)} \end{aligned}$$

From Lemma 6.2.1 we know $E[\epsilon] = 0$ gives:

$$Var(\epsilon_i) = (L_{ii})^{-2}\left(\mu_i\left(1-\mu_i\right) - \sum_{j=1}^{i-1}\sum_{k=1}^{i-1} L_{ij}L_{ik}Cov(\epsilon_j, \epsilon_k)\right), \quad \text{(A.2.9)}$$

where $Cov(\epsilon_j, \epsilon_k) = 0$ for $j \neq k$ so:

$$Var(\epsilon_i) = (L_{ii})^{-2}\left(\mu_i(1-\mu_i) - \sum_{j=1}^{i-1} L_{ij}^2 Var(\epsilon_j)\right), \quad \text{(A.2.10)}$$

For ϵ_1 we can write the variance as:

$$\begin{aligned} Var(\epsilon_1) &= \mu_1\left(L_{11}^{-1}(1-\mu_1)\right)^2 + (1-\mu_1)\left(-L_{11}^{-1}\mu_1\right)^2 \\ &= L_{11}^{-2}\mu_1(1-\mu_1)^2 + L_{11}^{-2}\mu_1^2(1-\mu_1) \\ &= L_{11}^{-2}\mu_1(1-\mu_1)(1-\mu_1+\mu_1) \\ &= L_{11}^{-2}\left(\mu_1(1-\mu_1)\right) \quad \text{(A.2.11)} \end{aligned}$$

The random variable Z_1 has distribution $\mathbf{Z_1} \sim Bernoulli(\mu_1)$ so has variance $\Sigma_{11} = L_{11}^2 = \mu_1(1-\mu_1)$ and (A.2.11) simplifies to:

$$Var(\epsilon_1) = L_{11}^{-2}L_{11}^2 = 1. \quad \text{(A.2.12)}$$

We can then show that all ϵ_i have $Var(\epsilon_1) = 1$ by induction. Assume $Var(\epsilon_k) = 1$ for

$k = 1, \ldots, i-1$, then (A.2.10) is equivalent to:

$$Var(\epsilon_i) = (L_{ii})^{-2} \left(\mu_i \left(1 - \mu_i\right) - \sum_{j=1}^{i-1} L_{ij}^2 \right). \tag{A.2.13}$$

The random variable Z_i has distribution $Z_i \sim Bernoulli(\mu_i)$ and so has variance $Var(Z_i) = \mu_i(1-\mu_i)$ which is written in terms of the Cholesky decomposition as:

$$Var(Z_i) = \mu_i(1-\mu_i) = \sum_{j=1}^{i-1} L_{ij}^2 + L_{ii}^2. \tag{A.2.14}$$

Substituting into (A.2.13) we get:

$$\begin{aligned} Var(\epsilon_i) &= (L_{ii})^{-2} \left(\sum_{j=1}^{i-1} L_{ij}^2 + L_{ii}^2 - \sum_{j=1}^{i-1} L_{ij}^2 \right), \\ &= (L_{ii})^{-2} L_{ii}^2, \\ &= 1. \end{aligned} \tag{A.2.15}$$

Hence, $Var(\epsilon_i) = 1$ and the proof holds by induction. □

A.2.2 Third Centralised Moment Proofs

Proof (Proof of Lemma 6.3.3)**.** Firstly, we show that the if there is a single additional random variable between Z_j and Z_k in the ordering, the conditional expectation $E[Z_k \mid \mathbf{Z}_{1:j}]$ is not altered using the BL method. Using Lemma 6.3.1 we can write the conditional expectation of $E[Z_k \mid \mathbf{Z}_{1:j+1}]$ where $k = j+2$ as:

$$E[Z_k \mid \mathbf{Z}_{1:j+1}] = \mu_k - \sum_{n=1}^{j+1} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n) + \sum_{n=1}^{j+1} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n). \tag{A.2.16}$$

This conditional expectation can be written as a function of Z_{j+1} where:

$$E[Z_k \mid \mathbf{Z}_{1:j+1}] = \mu_k - \sum_{n=1}^{j+1} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n) + \sum_{n=1}^{j} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n)$$
$$+ L_{k,j+1}(L_{j+1,j+1})^{-1}(Z_{j+1} - \mu_{j+1}) \quad \text{(A.2.17)}$$
$$= a + b(Z_{j+1} - \mu_{j+1}). \quad \text{(A.2.18)}$$

The conditional expectation $E[Z_k \mid \mathbf{Z}_{1:j}]$ is found by taking the expectation of A.2.18 over Z_{j+1}. If $p = E[Z_{j+1} \mid Z_{1:j}]$ this is:

$$E[E[Z_k \mid \mathbf{Z}_{1:j+1}]] = pE[Z_k \mid \mathbf{Z}_{1:j}, Z_{j+1} = 1] + (1-p)E[Z_k \mid \mathbf{Z}_{1:j}, Z_{j+1} = 0] \quad \text{(A.2.19)}$$
$$= p(a + b(Z_{j+1} - \mu_{j+1})) + (1-p)(a - b\mu_{j+1}) \quad \text{(A.2.20)}$$
$$= b(p - \mu_{j+1}) + a \quad \text{(A.2.21)}$$

Substituting the values for a, b and p back into (A.2.21) gives:

$$E[E[Z_k \mid \mathbf{Z}_{1:j+1}]] = L_{k,j+1}(L_{j+1,j+1})^{-1}\left(E[Z_{j+1} \mid \mathbf{Z}_{1:j}] - \mu_{j+1}\right)$$
$$+ \mu_k - \sum_{n=1}^{j+1} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n)$$
$$+ \sum_{n=1}^{j} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n)$$

Taking the jth term out of the summation:

$$E[E[Z_k \mid \mathbf{Z}_{1:j+1}]] = L_{k,j+1}(L_{j+1,j+1})^{-1}\left(E[Z_{j+1} \mid \mathbf{Z}_{1:j}] - \mu_{j+1}\right)$$
$$+ \mu_k - \sum_{n=1}^{j} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n)$$
$$- L_{k,j+1}(L_{j+1,j+1})^{-1}\left(E[Z_{j+1} \mid \mathbf{Z}_{1:j}] - \mu_{j+1}\right)$$
$$+ \sum_{n=1}^{j} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n)$$

which cancels to give:

$$=\mu_k - \sum_{n=1}^{j} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n) + \sum_{n=1}^{j} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n) \tag{A.2.22}$$

$$=E[Z_k \mid \mathbf{Z}_{1:j}] \tag{A.2.23}$$

using Lemma 6.3.2. Hence, the BL conditional expectation does not depend on whether a single additional random variable is in the ordering between Z_j and Z_k.

Next we show that the property holds when Z_k is the mth random variable or the $m+1$th random variable. Using Lemma 6.3.1, $k = m+1$, and the random variables are simulated numerically, the BL conditional expectation is:

$$E[Z_k \mid \mathbf{Z}_{1:m}] = \mu_k - \sum_{n=1}^{m} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n) + \sum_{n=1}^{m} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n). \tag{A.2.24}$$

This can be written as a function of Z_m where:

$$\begin{aligned} E[Z_k \mid \mathbf{Z}_{1:j+1}] =& \mu_k - \sum_{n=1}^{m} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n) + \sum_{n=1}^{m-1} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n) \\ &+ L_{km}L_{mm}^{-1}(Z_m - \mu_m) \qquad \text{(A.2.25)} \\ =& a + b(Z_m - \mu_m). \qquad \text{(A.2.26)} \end{aligned}$$

The conditional expectation $E[Z_k \mid \mathbf{Z}_{1:m-1}]$ is then found by taking the expectation over Z_m. If $p = E[Z_m \mid Z_{1:m-1}]$ this is:

$$\begin{aligned} E[E[Z_k \mid \mathbf{Z}_{1:m}]] =& pE[Z_k \mid \mathbf{Z}_{1:m-1}, Z_m = 1] + (1-p)E[Z_k \mid \mathbf{Z}_{1:m-1}, Z_m = 0] \qquad \text{(A.2.27)} \\ =& b(p - \mu_m) + a \qquad \text{(A.2.28)} \end{aligned}$$

Substituting the values for a, b and p back into (A.2.28) gives:

$$\begin{aligned}E[E[Z_k \mid \mathbf{Z}_{1:m}]] =& L_{km}(L_{mm})^{-1}\left(E[Z_m \mid \mathbf{Z}_{1:m-1}] - \mu_m\right) \\ &+ \mu_k - \sum_{n=1}^{m} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n) \quad + \sum_{n=1}^{m-1} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n)\end{aligned}$$

which simplifies to:

$$E[E[Z_k \mid \mathbf{Z}_{1:m}]] = \mu_k - \sum_{n=1}^{m-1} L_{kn}(L_{nn})^{-1}(E[Z_n \mid \mathbf{Z}_{1:n-1}] - \mu_n) + \sum_{n=1}^{m-1} L_{kn}(L_{nn})^{-1}(Z_n - \mu_n) \tag{A.2.29}$$

$$= E[Z_k \mid \mathbf{Z}_{1:m-1}] \tag{A.2.30}$$

using Lemma 6.3.2. Hence, if the conditional expectation is the same when k is the $m+1$th random variable and when it is the mth and it is the same when it is the $j+1$th random variable as when it is $j+2$th, then we can show that:

$$\begin{aligned}E[Z_k \mid \mathbf{Z}_{1:m}] =& E[\cdots E[E[E[Z_k \mid \mathbf{Z}_{1:m}]]]\cdots] \\ =& E[\cdots E[E[Z_k \mid \mathbf{Z}_{1:m-1}]]\cdots] = \ldots = E[E[Z_k \mid \mathbf{Z}_{1:j+1}]] \\ =& E[Z_k \mid \mathbf{Z}_{1:j+1}]\end{aligned} \tag{A.2.31}$$

Hence, the proof holds by induction. □

Bibliography

Agrawal, S. and Goyal, N. (2012). Analysis of thompson sampling for the multi-armed bandit problem. In *COLT*, pages 39–1.

Amir, E. (2001). Efficient approximation for triangulation of minimum treewidth. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 7–15. Morgan Kaufmann Publishers Inc.

Andersen, M. S., Dahl, J., and Vandenberghe, L. (2013). CVXOPT: A python package for convex optimization, version 1.1.6. *Available at* `cvxopt.org`.

Andrieu, C., Doucet, A., and Holenstein, R. (2010). Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(3):269–342.

Arnborg, S., Corneil, D. G., and Proskurowski, A. (1987). Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic Discrete Methods*, 8(2):277–284.

Attias, H. (1999). Inferring parameters and structure of latent variable models by variational Bayes. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 21–30. Morgan Kaufmann Publishers Inc.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256.

Baxter, R. J. (1982). *Exactly solved models in statistical mechanics*. Elsevier.

Becker, A. and Geiger, D. (1996). A sufficiently fast algorithm for finding close to optimal junction trees. In *Proceedings of the twelfth international conference on uncertainty in artificial intelligence*, pages 81–89. Morgan Kaufmann Publishers Inc.

Bellman, R. (1954). The theory of dynamic programming. Technical report, DTIC Document.

Bengtsson, T., Bickel, P., Li, B., et al. (2008). Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. In *Probability and statistics: Essays in honor of David A. Freedman*, pages 316–334. Institute of Mathematical Statistics.

Berrou, C. and Glavieux, A. (1996). Near optimum error correcting coding and decoding: Turbo-codes. *IEEE Transactions on communications*, 44(10):1261–1271.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 192–236.

Beskos, A., Crisan, D., Jasra, A., et al. (2014a). On the stability of sequential Monte Carlo methods in high dimensions. *The Annals of Applied Probability*, 24(4):1396–1445.

Beskos, A., Crisan, D., Jasra, A., Kamatani, K., and Zhou, Y. (2014b). A stable particle filter in high-dimensions. *arXiv preprint arXiv:1412.3501*.

Bethe, H. A. (1935). Statistical theory of superlattices. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 150(871):552–575.

Bishop, C. M. et al. (2006). *Pattern recognition and machine learning*, volume 4. springer New York.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2016). Variational inference: A review for statisticians. *arXiv preprint arXiv:1601.00670*.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge university press.

Briggs, J., Dowd, M., and Meyer, R. (2013). Data assimilation for large-scale spatio-temporal systems using a location particle smoother. *Environmetrics*, 24(2):81–97.

Brown, D. B. and Smith, J. E. (2013). Optimal sequential exploration: Bandits, clairvoyants, and wildcats. *Operations research*, 61(3):644–665.

Burnetas, A. N. and Katehakis, M. N. (1996). Optimal adaptive policies for sequential allocation problems. *Advances in Applied Mathematics*, 17(2):122–142.

Caimo, A. and Friel, N. (2011). Bayesian inference for exponential random graph models. *Social Networks*, 33(1):41–55.

Cappé, O., Godsill, S. J., and Moulines, E. (2007). An overview of existing methods and recent advances in sequential Monte Carlo. *Proceedings of the IEEE*, 95(5):899–924.

Caron, S. and Bhagat, S. (2013). Mixing bandits: A recipe for improved cold-start recommendations in a social network. In *Proceedings of the 7th Workshop on Social Network Mining and Analysis*, page 11. ACM.

Carpentier, A. and Valko, M. (2016). Revealing graph bandits for maximizing local influence. In *International Conference on Artificial Intelligence and Statistics*, pages 10–18.

Cavaliere, F., Mandal, P., and Barnes, C. (2005). Can Company E-Mail Avoid Becoming Evidence Mail? *Proceedings of the 2006 Southwest Decision Sciences Institute.*

Cesa-Bianchi, N., Gentile, C., and Zappella, G. (2013). A gang of bandits. In *Advances in Neural Information Processing Systems*, pages 737–745.

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89(3):539–552.

Chopin, N. (2004). Central limit theorem for sequential Monte Carlo methods and its application to Bayesian inference. *Annals of statistics*, pages 2385–2411.

Computational Analysis of Social and Organizational Systems (2004). Tanzania embassy CT. Available at: `http://www.casos.cs.cmu.edu/computational_tools/datasets/internal/embassy/index11.php`.

Craig, P. S., Goldstein, M., Seheult, A., and Smith, J. (1998). Constructing partial prior specifications for models of complex physical systems. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(1):37–53.

Craig, P. S., Goldstein, M., Seheult, A. H., and Smith, J. A. (1997). Pressure matching for hydrocarbon reservoirs: a case study in the use of Bayes linear strategies for large computer experiments. In *Case studies in Bayesian statistics*, pages 37–93. Springer.

De Finetti, B. (1974). Theory of probability, volume I. *Bull. Amer. Math. Soc*, 2(9904):14.

Diggle, P., Zeger, S., Liang, K., and Heagerty, P. (2002). *Analysis of longitudinal data*. Oxford University Press.

Dimitrov, N. B., Kress, M., and Nevo, Y. (2016). Finding the needles in the haystack: efficient intelligence processing. *Journal of the Operational Research Society*, 67(6):801–812.

Douc, R. and Moulines, E. (2007). Limit theorems for weighted samples with applications to sequential monte carlo methods. In *ESAIM: Proceedings*, volume 19, pages 101–107. EDP Sciences.

Doucet, A., Godsill, S., and Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and computing*, 10(3):197–208.

Doucet, A. and Johansen, A. M. (2009). A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3.

Draper, D. L. and Hanks, S. (1994). Localized partial evaluation of belief networks. In *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, pages 170–177. Morgan Kaufmann Publishers Inc.

Duyvesteyn, I., de Jong, B., and van Reijn, J. (2014). *The Future of Intelligence: Challenges in the 21st Century*. Routledge.

Ellis, D. R. (2013). Algorithms for efficient intelligence collection. Technical report, DTIC Document.

Emrich, L. J. and Piedmonte, M. R. (1991). A method for generating high-dimensional multivariate binary variates. *The American Statistician*, 45(4):302–304.

Everitt, R. G. (2012). Bayesian parameter estimation for latent Markov random fields and social networks. *Journal of Computational and graphical Statistics*, 21(4):940–960.

Farrow, M. and Goldstein, M. (2006). Trade-off sensitive experimental design: a multi-criterion, decision theoretic, Bayes linear approach. *Journal of Statistical Planning and Inference*, 136(2):498–526.

Fearnhead, P. and Clifford, P. (2003). On-line inference for hidden Markov models via particle filters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(4):887–899.

Flynn, N. and Kahn, R. (2003). *E-Mail rules: a business guide to managing policies, security, and legal issues for E-mail and digital communication.* American Management Assoc., Inc.

Fowler, J. H., Johnson, T. R., Spriggs, J. F., Jeon, S., and Wahlbeck, P. J. (2007). Network analysis and the law: Measuring the legal importance of precedents at the US Supreme Court. *Political Analysis*, 15(3):324–346.

Frazier, P. I. (2011). Learning with dynamic programming. *Wiley Encyclopedia of Operations Research and Management Science*.

Friel, N., Pettitt, A., Reeves, R., and Wit, E. (2009). Bayesian inference in hidden Markov random fields for binary data defined on large lattices. *Journal of Computational and Graphical Statistics*, 18(2):243–261.

Gange, S. J. (1995). Generating multivariate categorical variates using the iterative proportional fitting algorithm. *The American Statistician*, 49(2):134–138.

Garivier, A. and Cappé, O. (2011). The KL-UCB algorithm for bounded stochastic bandits and beyond. In *COLT*, pages 359–376.

Geman, S. and Geman, D. (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on pattern analysis and machine intelligence*, 6:721–741.

George, E. I. and McCulloch, R. E. (1997). Approaches for Bayesian variable selection. *Statistica sinica*, pages 339–373.

Ghahramani, Z. (2001). An introduction to hidden Markov models and Bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(01):9–42.

Gilks, W. R., Thomas, A., and Spiegelhalter, D. J. (1994). A language and program for complex Bayesian modelling. *The Statistician*, pages 169–177.

Gittins, J. C. and Jones, D. M. (1979). A dynamic allocation index for the discounted multiarmed bandit problem. *Biometrika*, pages 561–565.

Goldstein, M. and Rougier, J. (2009). Reified Bayesian modelling and inference for physical systems. *Journal of Statistical Planning and Inference*, 139(3):1221–1239.

Goldstein, M. and Wooff, D. (2007). *Bayes linear statistics, theory and methods*, volume 716. John Wiley & Sons.

Gordon, N. J., Salmond, D. J., and Smith, A. F. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113. IET.

Gore, V. K. and Jerrum, M. R. (1999). The Swendsen–Wang process does not always mix rapidly. *Journal of Statistical Physics*, 97(1-2):67–86.

Gorman, S. (2008). Probe on Christmas plot lists failures. *The Wall Street Journal*. Available at: `http://www.wsj.com/articles/SB10001424052748704912004575252861734663130`.

Hagberg, A. A., Schult, D. A., and Swart, P. J. (2008). Exploring network structure, dynamics, and function using NetworkX. In *Proceedings of the 7th Python in Science Conference (SciPy2008)*, pages 11–15, Pasadena, CA USA.

Hammersley, J. M. and Clifford, P. (1971). Markov fields on finite graphs and lattices.

Hammersley, J. M. and Morton, K. W. (1954). Poor man's Monte Carlo. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 23–38.

Hamze, F. and de Freitas, N. (2005). Hot coupling: A particle approach to inference and normalization on pairwise undirected graphs of arbitrary topology. *Advances in neural information processing systems*, 18:1–8.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

Henrion, M. (1991). Search-based methods to bound diagnostic probabilities in very large belief nets. In *Proceedings of the Seventh conference on Uncertainty in Artificial Intelligence*, pages 142–150. Morgan Kaufmann Publishers Inc.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. W. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14(1):1303–1347.

Hughbank, R. J. and Githens, D. (2010). Intelligence and its role in protecting against terrorism. *Journal of Strategic Security*, 3(1):31.

Isard, M. (2003). PAMPAS: Real-valued graphical models for computer vision. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–613. IEEE.

Jordan, M. I. (2004). Graphical models. *Statistical Science*, pages 140–155.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. (1999). An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233.

Kaelbling, L. P. (1993). *Learning in embedded systems*. MIT press.

Kang, S.-H. and Jung, S.-H. (2001). Generating correlated binary variables with complete specification of the joint distribution. *Biometrical Journal*, 43(3):263–269.

Kaufmann, E., Cappé, O., and Garivier, A. (2012a). On Bayesian Upper Confidence Bounds for Bandit Problems. In *AISTATS*, pages 592–600.

Kaufmann, E., Korda, N., and Munos, R. (2012b). Thompson sampling: An asymptotically optimal finite-time analysis. In *International Conference on Algorithmic Learning Theory*, pages 199–213. Springer.

Kim, S.-H. and Nelson, B. L. (2006). Selecting the best system. *Handbooks in operations research and management science*, 13:501–534.

Kjærulff, U. (1994). Reduction of computational complexity in Bayesian networks through removal of weak dependences. In *Proceedings of the tenth international conference on uncertainty in artificial intelligence*, pages 374–382. Morgan Kaufmann Publishers Inc.

Koller, D. and Friedman, N. (2009). *Probabilistic graphical models: principles and techniques*. MIT press.

Kong, A., Liu, J. S., and Wong, W. H. (1994). Sequential imputations and Bayesian missing data problems. *Journal of the American statistical association*, 89(425):278–288.

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50.

Kschischang, F. R., Frey, B. J., and Loeliger, H.-A. (2001). Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519.

Lai, T. L. (1987). Adaptive treatment allocation and the multi-armed bandit problem. *The Annals of Statistics*, pages 1091–1114.

Lai, T. L. and Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 6(1):4–22.

Lauritzen, S. L. and Spiegelhalter, D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 157–224.

Lee, A. (1993). Generating random binary deviates having fixed marginal distributions and specified degrees of association. *The American Statistician*, 47(3):209–215.

Leisink, M. A. and Kappen, H. J. (2002). Computer generated higher order expansions. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 293–300. Morgan Kaufmann Publishers Inc.

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media.

Loh, P.-L., Wainwright, M. J., et al. (2012). Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses. In *NIPS*, pages 2096–2104.

Loh, P.-L., Wainwright, M. J., et al. (2013). Structure estimation for discrete graphical models: Generalized covariance matrices and their inverses. *The Annals of Statistics*, 41(6):3022–3049.

Luce, R. D. (2005). *Individual choice behavior: A theoretical analysis*. Courier Corporation.

Maneva, E., Mossel, E., and Wainwright, M. J. (2007). A new look at survey propagation and its generalizations. *Journal of the ACM (JACM)*, 54(4):17.

Martinelli, G., Eidsvik, J., and Hauge, R. (2013). Dynamic decision making for graphical models applied to oil exploration. *European Journal of Operational Research*, 230(3):688–702.

May, B. C., Korda, N., Lee, A., and Leslie, D. S. (2012). Optimistic Bayesian sampling in contextual-bandit problems. *Journal of Machine Learning Research*, 13(Jun):2069–2106.

McEliece, R. J., MacKay, D. J. C., and Cheng, J.-F. (1998). Turbo decoding as an instance of Pearl's belief propagation algorithm. *IEEE Journal on selected areas in communications*, 16(2):140–152.

McGinnis, J. O. and Pearce, R. G. (2014). The great disruption: How machine intelligence will transform the role of lawyers in the delivery of legal services.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092.

Miller, R. A., Pople Jr, H. E., and Myers, J. D. (1982). Internist-I, an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine*, 307(8):468–476.

Minka, T. P. (2001). *A family of algorithms for approximate Bayesian inference*. PhD thesis, Massachusetts Institute of Technology.

Modarres, R. (2011). High-dimensional generation of Bernoulli random vectors. *Statistics & Probability Letters*, 81(8):1136–1142.

Møller, J., Pettitt, A. N., Reeves, R., and Berthelsen, K. K. (2006). An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants. *Biometrika*, 93(2):451–458.

Mullin, J. (2016). How Oracle made its case against Google, in pictures. *Ars Technica*. [Online; posted 25-May-2016].

Murphy, K. P., Weiss, Y., and Jordan, M. I. (1999). Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 467–475. Morgan Kaufmann Publishers Inc.

Murray, I., Ghahramani, Z., and MacKay, D. (2012). MCMC for doubly-intractable distributions. *arXiv preprint arXiv:1206.6848*.

Naesseth, C. A., Lindsten, F., and Schön, T. B. (2014). Sequential Monte Carlo for graphical models. In *Advances in Neural Information Processing Systems*, pages 1862–1870.

Naesseth, C. A., Lindsten, F., and Schön, T. B. (2016). High-dimensional filtering using nested sequential Monte Carlo. *arXiv preprint arXiv:1612.09162*.

Needham, C. J., Bradford, J. R., Bulpitt, A. J., and Westhead, D. R. (2007). A primer on learning in Bayesian networks for computational biology. *PLoS Comput Biol*, 3(8):e129.

Nemhauser, G. L. and Wolsey, L. A. (1988). Integer programming and combinatorial optimization. *Wiley, Chichester. GL Nemhauser, MWP Savelsbergh, GS Sigismondi (1992)*.

Constraint Classification for Mixed Integer Programming Formulations. COAL Bulletin, 20:8–12.

Nevo, Y. (2011). Information selection in intelligence processing. Technical report, DTIC Document.

Nikovski, D. (2000). Constructing Bayesian networks for medical diagnosis from incomplete and partially correct statistics. *IEEE Transactions on Knowledge and Data Engineering*, 12(4):509–516.

Pandey, S., Chakrabarti, D., and Agarwal, D. (2007). Multi-armed bandit problems with dependent arms. In *Proceedings of the 24th international conference on Machine learning*, pages 721–728. ACM.

Pearl, J. (1982). Reverend Bayes on inference engines: A distributed hierarchical approach. In *AAAI*, pages 133–136.

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599.

Powell, W. B. and Ryzhov, I. O. (2012). *Optimal learning*, volume 841. John Wiley & Sons.

Punskaya, E., Doucet, A., and Fitzgerald, W. J. (2002). On the use and misuse of particle filtering in digital communications. In *Signal Processing Conference, 2002 11th European*, pages 1–4. IEEE.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Qaqish, B. F. (2003). A family of multivariate binary distributions for simulating correlated binary variables with specified marginal means and correlations. *Biometrika*, 90(2):455–463.

Rebeschini, P., Van Handel, R., et al. (2015). Can local particle filters beat the curse of dimensionality? *The Annals of Applied Probability*, 25(5):2809–2866.

Ripley, B. D. (2005). *Spatial statistics*, volume 575. John Wiley & Sons.

Robbins, H. (1952). Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535.

Robert, C. and Casella, G. (2013). *Monte Carlo statistical methods*. Springer Science & Business Media.

Robert, C. P. (2004). *Monte Carlo methods*. Wiley Online Library.

Robertson, N. and Seymour, P. D. (1986). Graph minors. II. Algorithmic aspects of tree-width. *Journal of algorithms*, 7(3):309–322.

Rubinstein, R. (1999). The cross-entropy method for combinatorial and continuous optimization. *Methodology and computing in applied probability*, 1(2):127–190.

Rusmevichientong, P. and Tsitsiklis, J. N. (2010). Linearly parameterized bandits. *Mathematics of Operations Research*, 35(2):395–411.

Ryzhov, I. O., Powell, W. B., and Frazier, P. I. (2012). The knowledge gradient algorithm for a general class of online learning problems. *Operations Research*, 60(1):180–195.

Saul, L. K. and Jordan, M. I. (1996). Exploiting tractable substructures in intractable networks. *Advances in neural information processing systems*, pages 486–492.

Scott, S. L. (2010). A modern Bayesian look at the multi-armed bandit. *Applied Stochastic Models in Business and Industry*, 26(6):639–658.

Select Committee on Intelligence, United State Senate (2010). Attempted terrorist attack on Northwest Airlines Flight 253. Available at: `http://www.intelligence.senate.gov/pdfs/111199.pdf`.

Shachter, R. D., Andersen, S. K., and Szolovits, P. (1994). Global conditioning for probabilistic inference in belief networks. In *Proceedings of the Tenth international conference on Uncertainty in artificial intelligence*, pages 514–522. Morgan Kaufmann Publishers Inc.

Shafer, G. R. and Shenoy, P. P. (1990). Probability propagation. *Annals of Mathematics and Artificial Intelligence*, 2(1-4):327–351.

Smith, D. K. (1991). *Dynamic programming: a practical introduction*. Prentice Hall.

Sudderth, E. B., Ihler, A. T., Isard, M., Freeman, W. T., and Willsky, A. S. (2010). Nonparametric belief propagation. *Communications of the ACM*, 53(10):95–103.

Sun, J., Zheng, N.-N., and Shum, H.-Y. (2003). Stereo matching using belief propagation. *IEEE Transactions on pattern analysis and machine intelligence*, 25(7):787–800.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Tatikonda, S. C. and Jordan, M. I. (2002). Loopy belief propagation and Gibbs measures. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 493–500. Morgan Kaufmann Publishers Inc.

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294.

Tsitsiklis, J. N. (1994). A short proof of the gittins index theorem. *The Annals of Applied Probability*, pages 194–199.

Tugnait, J. K. (1982). Detection and estimation for abruptly changing systems. *Automatica*, 18(5):607–615.

Vermorel, J. and Mohri, M. (2005). Multi-armed bandit algorithms and empirical evaluation. In *European conference on machine learning*, pages 437–448. Springer.

Vernon, I., Goldstein, M., Bower, R. G., et al. (2010). Galaxy formation: a Bayesian uncertainty analysis. *Bayesian Analysis*, 5(4):619–669.

Viterbi, A. (1967). Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269.

Volonino, L. (2003). Electronic evidence and computer forensics. *Communications of the Association for Information Systems*, 12(1):27.

Wainwright, M. J. and Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305.

Wang, C.-C., Kulkarni, S. R., and Poor, H. V. (2005). Bandit problems with side observations. *IEEE Transactions on Automatic Control*, 50(3):338–355.

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. PhD thesis, University of Cambridge England.

Whittaker, J. (2009). *Graphical models in applied multivariate statistics*. Wiley Publishing.

Whittle, P. (1982). *Optimization over time*. John Wiley & Sons, Inc.

Whittle, P. (2012). *Probability via expectation*. Springer Science & Business Media.

Williamson, D., Goldstein, M., Allison, L., Blaker, A., Challenor, P., Jackson, L., and Yamazaki, K. (2013). History matching for exploring and reducing climate model parameter space using observations and a large perturbed physics ensemble. *Climate dynamics*, 41(7-8):1703–1729.

Winn, J. and Bishop, C. M. (2005). Variational message passing. *Journal of Machine Learning Research*, 6(Apr):661–694.

Yedidia, J. S., Freeman, W. T., and Weiss, Y. (2003). Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239.

Zaritskii, V., Svetnik, V., and Šimelevič, L. (1975). Monte-Carlo technique in problems of optimal information processing. *Avtomatika i Telemekhanika*, 12:95–103.

Zhang, N. L. and Poole, D. (1994). A simple approach to Bayesian network computations. In *Proc. of the Tenth Canadian Conference on Artificial Intelligence*.