

LOSSLESS CODING OF MARKOV RANDOM FIELDS  
WITH COMPLEX CLIQUES

by

SZU KUAN STEVEN WU

A thesis submitted to the  
Department of Mathematics and Statistics  
in conformity with the requirements for  
the degree of Master of Applied Science

Queen's University  
Kingston, Ontario, Canada

August 2013

Copyright © Szu Kuan Steven Wu, 2013

# Abstract

The topic of Markov Random Fields (MRFs) has been well studied in the past, and has found practical use in various image processing, and machine learning applications. Where coding is concerned, MRF specific schemes have been largely unexplored. In this thesis, an overview is given of recent developments and challenges in the lossless coding of MRFs. Specifically, we concentrate on difficulties caused by computational intractability due to the partition function of the MRF. One proposed solution to this problem is to segment the MRF with a cutset, and encode the components separately. Using this method, arithmetic coding is possible via the Belief Propagation (BP) algorithm. We consider two cases of the BP algorithm: MRFs with only simple cliques, and MRFs with complex cliques. In the latter case, we study a minimum radius condition requirement for ensuring that all cliques are accounted for during coding. This condition also simplifies the process of conditioning on observed sites. Finally, using these results, we develop a systematic procedure of clustering and choosing cutsets.

# Table of Contents

<b>Abstract</b>	<b>i</b>
<b>Table of Contents</b>	<b>ii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Problem and Contributions . . . . .	3
1.3 Overview of Thesis . . . . .	4
<b>Chapter 2:</b>	
<b>Background . . . . .</b>	<b>5</b>
2.1 Graphs . . . . .	5
2.2 Markov Random Fields (MRF) . . . . .	7
2.2.1 Some Notational Conventions . . . . .	8
2.2.2 Neighbourhoods, Cliques, and Potentials . . . . .	9
2.2.3 MRF's and Gibbs Distributions . . . . .	11
2.2.4 Useful Properties and Theorems . . . . .	12
2.2.5 Ising Model . . . . .	15
2.3 Homogenous and Nonhomogenous Markov Chains . . . . .	17
2.3.1 Overview of Basic Notions . . . . .	17
2.3.2 Convergence of Homogeneous Markov Chains . . . . .	19
2.3.3 Gibbs Sampler . . . . .	21
2.4 Information Theory and Source Coding . . . . .	25
2.4.1 Overview of Basic Information Theory . . . . .	25
2.4.2 Lossless Source Coding . . . . .	27
2.4.3 Shannon-Fano-Elias Coding . . . . .	30
2.4.4 Universal and Arithmetic Coding . . . . .	31
2.5 Estimation and Inference . . . . .	33
2.5.1 Statistics and Moments . . . . .	34
2.5.2 Least Squares Estimation . . . . .	36
2.5.3 Maximum Likelihood Estimation . . . . .	37

2.5.4	Belief Propagation . . . . .	39
2.5.5	Clustering . . . . .	41
2.6	Coding of MRF's . . . . .	42
2.6.1	Entropy of MRFs . . . . .	42
2.6.2	Reduced MRFs . . . . .	43
2.6.3	Reduced Cutset Coding (RCC) . . . . .	46
<b>Chapter 3:</b>		
	<b>Learning and Precoding . . . . .</b>	<b>49</b>
3.1	Learned RCC . . . . .	49
3.1.1	MRF and Cutset Specifications . . . . .	50
3.1.2	Learning . . . . .	51
3.1.3	Encoding Parameter, Cutset, and Components . . . . .	51
3.1.4	Some Remarks About the Decoder . . . . .	52
3.2	Estimation and Encoding . . . . .	53
3.2.1	Using LS Estimation . . . . .	54
3.2.2	Using ML Estimation . . . . .	55
3.2.3	Parameter Sensitivity . . . . .	64
<b>Chapter 4:</b>		
	<b>MRFs with Complex Cliques . . . . .</b>	<b>69</b>
4.1	Some Preliminaries . . . . .	70
4.1.1	Some Definitions . . . . .	70
4.1.2	Cluster Graphs . . . . .	71
4.2	Useful Lemmas and Corollaries . . . . .	72
4.3	Clustering and Belief Propagation for MRF's with Complex Cliques . . . . .	76
4.3.1	Beliefs, Messages, and Recursion . . . . .	77
4.4	Conditional Beliefs and Cutset Coding . . . . .	82
4.4.1	Conditional Beliefs . . . . .	82
4.4.2	Conditional Belief Propagation . . . . .	89
4.5	Classifying Cliques and Potentials . . . . .	94
4.5.1	Clique Types . . . . .	94
4.5.2	Linear Parameter Potentials . . . . .	96
4.6	Estimation . . . . .	97
4.6.1	Least Squares . . . . .	97
4.6.2	MLE . . . . .	99
<b>Chapter 5:</b>		
	<b>Precoding with Complex Cliques . . . . .</b>	<b>105</b>
5.1	Minimum Radius Clustering . . . . .	105
5.2	Results and Discussion . . . . .	108

5.2.1	Some Results . . . . .	109
5.2.2	Discussion . . . . .	114
5.3	Future Work . . . . .	115
5.3.1	Identifying Cliques and Potentials . . . . .	115
5.3.2	Coding and Computational Considerations . . . . .	116
	<b>Bibliography . . . . .</b>	<b>117</b>

# Chapter 1

## Introduction

### 1.1 Motivation

The topic of data compression has been a well studied topic in mathematics, computer science, and engineering. Information Theory, fathered by Claude Shannon in 1948 [25], concerns itself with the quantification, storage, and communication of information. In particular, the study of source and channel coding provides interesting results regarding lossless and lossy compression. Though instructions on how to construct the codes themselves are not always explicit, the study of this topic is fruitful in finding bounds on compression rates (usually for a given source distribution or stochastic process). Given that stochastic processes are widely used in modelling various real world phenomena (images and audio to name a few), such bounds play a key role in understanding the limitations to compression and coding rates. Naturally, the study of coding methods capable of approaching these bounds for a given class of distributions has also become an important topic of interest. One mathematical structure from the exponential family, often used for image modelling, is the Markov

Random Field (MRF).

The study of MRFs was first inspired largely by statistical physics - more specifically, a motivating example called the Ising model. Physicist Ernest Ising, in 1925, used his model to describe the ferromagnetic interactions between iron atoms [15]. Under this model, individual iron atoms could be seen as nodes, and the effect of spin phases on neighbouring atoms could be seen as edges. Thus, many of the tools of graph theory have been applied to the study of MRFs as well. Bearing semblance to other probabilistic graphical structures such as belief and Bayesian networks, the MRF has found application in machine learning as well. This means tools and techniques such as Belief Propagation can be extended to the study of MRFs. However, it should be noted that the formal definition of MRFs allows for more complex structures beyond nodes and edges. These complex structures have great potential in image modelling, where they can be used to capture larger, more complex patterns. In more recent years, MRFs have gained popularity in image processing. Stuart and Donald Geman demonstrated, in 1984, the usefulness of MRFs in capturing textural patterns and features within images [12] by restoring noisy images using a MRF.

The usefulness of MRFs prompts the question of how one might encode such a structure optimally. While there are universal coding methods that function well for various sources, the MRF's unique structure makes it computationally difficult if not intractable. Nonetheless, the graphical nature of MRFs make them ideal candidates for application in distributed computing settings.

## 1.2 Problem and Contributions

In 2010, Reyes proposed both a lossy and a lossless method that sought to bridge the gap between MRF structures and conventional techniques [23]. Both techniques sought to break down the MRF structure into smaller, simpler structures, on which conventional techniques could be used. In the former, he proposed encoding part of the MRF using standard tools, and recovering the rest using MAP estimation. In the latter, he proposed using either clustering or local conditioning to attain an acyclic graph on which Belief Propagation could be used to compute coding distributions. He then suggested using arithmetic coding to encode the MRF.

In this thesis, we study a MRF encoding technique based on Reyes 2010. We improve the technique by introducing a learning phase to try and optimally match a MRF with a given image. Also, we choose not to adopt the local conditioning technique, as the method relies on intricately trimming edges and creating conditioned nodes to attain an acyclic graph. This is somewhat limiting as it relies heavily on node and edge structures and fails to function under MRFs where higher order structures are involved. We instead opt for the clustering technique, but generalize this technique to MRFs with larger neighbourhoods and clique structures.

We establish a framework for clustering on MRFs with complex clique structures, under which the traditional belief propagation technique (from a graphical approach) can function with minimal modifications. We also go on to show these necessary modifications and prove their functionality. The problem here is twofold. First, in establishing clusters that bear semblance to the usual nodes and edges, and second, ensuring that these clusters do not exhibit pathological behaviour. One should note that, under our framework, the clustering technique proposed by Reyes is a special



case where certain limitations are put upon the clique structures. For this reason, the generalized technique proposed in this thesis can be more advantageous and versatile in situations where a diverse set of clique structures are needed to capture desired features.

### 1.3 Overview of Thesis

In Chapter 2, we introduce basic notions of graph theory and Markov Random Fields. We also give short overviews of relevant results regarding Markov chains, information theory, and inference and estimation techniques including Belief Propagation. We end off the chapter with an overview of Reyes' lossless coding method known as Reduced Cutset Coding (RCC). Chapter 3 considers the problem adding the additional learning phase to the encoding process. In Chapter 4, we prove in detail the results that make the inference and estimation machinery function properly over MRFs with complex cliques. This is done in three steps. First, we go over some notions regarding cluster graphs and cluster edges. We then examine examples that exhibit pathological behaviour and prove the necessary results for avoiding such cases. Finally, we use these results to demonstrate the robustness of our inference and estimation tools. At last, Chapter 5 presents results using the techniques proposed in Chapter 4, concluding remarks, and suggestions for future work.

# Chapter 2

## Background

In this chapter, we present some background material on graphs, Markov Random Fields (MRF), information theory, and related literature. We start with some basic definitions and properties of graphs and MRFs. We then state important results regarding homogenous and inhomogenous Markov chains, Gibbs sampling, and the role these play in sampling MRFs. From here, we proceed by giving a quick overview of information theoretic results regarding source coding and arithmetic coding. This is then followed by a brief discussion on methods of inference on MRFs. We finish the chapter by reviewing related literature on lossless coding of MRFs.

### 2.1 Graphs

A **graph**  $G$  is a pair  $(V, E)$ , where  $V$  is a finite set of **nodes**, and  $E \subset V \times V$  is a set of pairs of nodes representing **edges** or connections between two nodes. The following are some additional definitions and terminology regarding graphs that are useful. We deal only with undirected graphs in this thesis.

- Two nodes  $i, j \in V$  are **adjacent** if  $(i, j) \in E$ .
- A **path** is a sequence of nodes  $(n_i)_{i=1}^N$  where successive pairs  $(n_m, n_{m+1})$  are adjacent, i.e.  $(n_m, n_{m+1}) \in E$ ,  $m = 1, \dots, N - 1$ .
- A **connected** graph is one where any two nodes  $i, j \in V$  can be joined by some path; the graph is disconnected otherwise.
- A **component**, denoted  $\mathcal{C}$ , of graph  $G$  is a maximal connected subset of  $V$ .
- A path where the first and last nodes coincide, and for which no other node repeats itself (no *backtracking*) is called a **cycle**.
- A graph with no cycles is **acyclic**. Connected acyclic graphs are often referred to as **trees**.
- For any set  $A \subset V$ , the **boundary**, denoted  $\partial A$ , of  $A$  is the set of nodes that are not in  $A$  but adjacent to at least one node in  $A$ . Similarly, the **surface** of  $A$ , denoted  $\gamma A$ , consist of nodes in  $A$  which are adjacent to at least 1 node in  $\partial A$ .
- For some subset  $A \subset V$ , let  $E_A \subset E$  be the set of edges where both ends of the edge reside in  $A$ , i.e.  $\{(i, j) \in E : i, j \in A\}$ . We call  $G_A = (A, E_A)$  the **subgraph** induced by  $A$ .
- For a connected graph  $G$  and some set  $L \subset V$ , if  $G_{L^c}$  is disconnected, then  $L$  is called a **cutset**. Let  $\{\mathcal{C}_n\}$  be the components of  $G_{L^c}$  and let  $A \subset \mathcal{C}_k$  for some  $k$ . For convenience, we introduce the notation  $G_{A \setminus L}$  to refer to  $\mathcal{C}_k$  (the component of  $G_{L^c}$  containing set  $A$ ).

- For a connected graph  $G$  and some edge  $e = (i, j) \in E$ , if removing  $e$  from  $E$  produces a disconnected graph, then  $e$  is called a **cut-edge**. By slightly adjusting the notation above, we identify components that result from removing cut-edge  $e$  from graph  $G$  by  $G_{i \setminus j}$  and  $G_{j \setminus i}$ . If this causes confusion, the reader can choose to use  $G_{i \setminus e}$  or  $G_{j \setminus (i, j)}$  instead.<sup>1</sup>

## 2.2 Markov Random Fields (MRF)

To start, we first provide the basic definition of a random field.

**Definition 2.2.1.** *Random Field*

Let  $S$  be a finite set of elements where individual elements, called **sites**, are denoted  $s$ . Let  $\Lambda$  be a finite set called the **phase space**. A **random field** is a collection  $X$  of random variables  $X(s)$  on sites  $s$  in  $S$  and taking values in  $\Lambda$ .

$$X = \{X(s)\}_{s \in S}.$$

The phase space  $\Lambda$  can also be called the **alphabet** on which the random variables  $X(s)$  assume values; this is familiar terminology from information theory. Note that the finiteness of  $\Lambda$  is not necessarily required, but in this thesis we will only consider ones which are finite in size. A good way of looking at a random field is to treat it as a random variable which takes values in the **configuration space**  $\Lambda^S$ .

---

<sup>1</sup>Note that *cutsets* and *cut-edges* are different. Both generate a disconnected graph when removed, but the former is a set of nodes, whereas the latter is an edge.

### 2.2.1 Some Notational Conventions

Individual configurations  $x \in \Lambda^S$  can be denoted  $x = \{x(s) : s \in S\}$ , where  $x(s) \in \Lambda$  for every  $s \in S$ . Also, we will use  $x_A$  to denote the configuration on a given subset  $A \subset S$ ; that is, for  $A \subset S$ ,

$$x_A = \{x(s) : s \in A\}$$

We will also use this notation in a consecutive manner to denote the configuration on a larger set composed of disjoint subsets. For instance, if we have,

$$A, B, C \subset S$$

$$A \cup B = C$$

$$A \cap B = \emptyset$$

then,

$$x_C = x_A x_B.$$

In cases where we have already defined some disjoint subsets of  $S$ , we will often adopt the right hand side to avoid having to explicitly define a new set. In general this notation will also be exercised when identifying a collection of random variables on a subset of sites, i.e.  $X_A$  refers to the random variables associated with the sites in  $A \subset S$ , and  $P(X_A = x_A)$  refers to the probability of some configuration on  $A$  with respect to the joint distribution of random variables  $X_A$ .

Finally, to explicitly specify a phase  $\lambda \in \Lambda$  on some site  $s$ , we will often use the notation  $\lambda_s$  as opposed to stating  $x_s = \lambda$ . For example, we can write  $x^* = \lambda_s x_{S \setminus s}$  to mean some configuration  $x^*$  that shares phase values with configuration  $x = x_s x_{S \setminus s}$

on the set  $S \setminus \{s\}$ , but holds the phase  $\lambda$  on site  $s$ .

## 2.2.2 Neighbourhoods, Cliques, and Potentials

The aforementioned definition of a random field is too general on its own; more structure is needed. Where coding is concerned, one will often ask if a source possesses some form of “Markov property”. This, along with some other characteristics, make finding a coding scheme tractable and/or easier. As we shall see soon, a Markov Random Field (MRF) possesses many of these useful properties we are looking for. But before we begin, first we must introduce some basic building blocks that help us define a MRF.

### Definition 2.2.2. Neighbourhoods

A **neighbourhood system** is the set  $N = \{N_s : s \in S\}$  of **neighbourhoods**  $N_s \subset S$  that satisfy the following,

$$(i) \quad s \notin N_s$$

$$(ii) \quad t \in N_s \iff s \in N_t.$$

The sites in  $N_s$  are referred to as the **neighbours** of  $s$ .

### Definition 2.2.3. Cliques

A **clique** is a subset  $C \subset S$  where any two distinct sites of  $C$  are mutual neighbours. Also, define any singleton  $\{s\}$  to be a clique.  $|C|$  is referred to as the **order** of the clique.

In this thesis, we will refer to cliques with order  $n \leq 2$  to be **simple cliques**, and those with order  $n > 2$  to be **complex cliques**.

**Definition 2.2.4.** *Potentials*

A potential relative to some neighbourhood system  $N$  is a function  $V_C : \Lambda^S \mapsto \mathbb{R} \cup \{+\infty\}$ ,  $C \subset S$ , where,

(i)  $V_C(x) \equiv 0$  if  $C$  is not a clique

(ii)  $\forall x, x^* \in \Lambda^S$  and  $\forall C \subset S$ , we have that,

$$(x_C = x^*_C) \implies (V_C(x) = V_C(x^*)).$$

For reasons which will be soon apparent, the collection of potential functions on all possible cliques is often referred to as the **Gibbs potential** relative to neighbourhood system  $N$ . Also, generally, the function  $V_C(x)$  will depend only on the configuration  $x_C$  of the sites within  $C$ . For the remainder of this thesis, we will be using the following notation interchangeably when appropriate,

$$V_C(x) = V_C(x_C).$$

In the case where say  $C = A \cup B$ , we may use  $V_C(x) = V_C(x_C) = V_C(x_A x_B)$

**Definition 2.2.5.** *Energy*

The energy is a function  $\varepsilon : \Lambda^S \mapsto \mathbb{R} \cup \{+\infty\}$ .

In particular, the energy function is said to be **derived from the Gibbs potential**  $\{V_C\}_{C \subset S}$  if,

$$\varepsilon(x) = \sum_{C \subset S} V_C(x).$$

### 2.2.3 MRF's and Gibbs Distributions

We now use the building blocks mentioned above to define the Gibbs distribution.

**Definition 2.2.6.** *Gibbs Distribution*

*The probability distribution,*

$$\pi_T(x) = \frac{1}{Q(T)} e^{-\frac{1}{T}\varepsilon(x)},$$

*where the energy function  $\varepsilon(x)$  is derived from a Gibbs potential,  $T > 0$ , is called a **Gibbs distribution**.*

Here,  $T$  denotes **temperature** and is useful in annealing; we will not deal with annealing in this thesis.  $Q(T)$  is the normalizing constant that takes into account the energy of all possible configurations on  $\Lambda^S$  and is often referred to as the **partition function**. It is given by,

$$Q(T) = \sum_{x \in \Lambda^S} \exp\left(-\frac{1}{T}\varepsilon(x)\right).$$

Later on, we will also denote the partition function by  $Q(\theta)$  to signify dependency on the parameter  $\theta$  that govern relevant potentials.

As evident from the expression for the Gibbs distribution, the probability of any given configuration is dictated by its energy, which in turn is a function of the cliques and potentials on the configuration. A random field whose distribution is a Gibbs Distribution is called a **Gibbs Field**.

In this thesis, we will not be conducting any temperature annealing and will usually just drop the  $T$  from the expression. This has the effect of assuming some



constant temperature; i.e., the term simply acts as a constant scaling to the energy of all configurations and we need not worry about its impact on the distribution.

Next, we formally define a MRF.

**Definition 2.2.7.** *Markov Random Field (MRF)*

A random field  $X$  is called a **Markov Random Field** with respect to a neighbourhood system  $N$  if  $\forall s \in S$  and  $x \in \Lambda^S$ , we have,

$$P(X_s = x_s | X_{S \setminus s} = x_{S \setminus s}) = P(X_s = x_s | X_{N_s} = x_{N_s}).$$

That is, the random variables  $X_s$  and  $X_{S \setminus (s \cup N_s)}$  are conditionally independent given  $X_{N_s}$ ; the distribution on site  $s$  is only directly influenced by the values of its neighbours.<sup>2</sup>

## 2.2.4 Useful Properties and Theorems

The following are some useful results about Gibbs distributions and MRFs. We refer to Brémaud 1999 [3] and Winkler 2003 [26] for proofs and in-depth treatment.

**Definition 2.2.8.** *Local Specification*

The **local characteristic** of a MRF at site  $s$  is given by,

$$\pi^s(x) = P(X_s = x_s | X_{N_s} = x_{N_s}),$$

where  $\pi^s : \Lambda^S \mapsto [0, 1]$ . The collection  $\{\pi^s\}_{s \in S}$  is referred to as the **local specification** of the MRF.

---

<sup>2</sup>We abuse notation a little bit and use  $s$  interchangeably with  $\{s\}$  to avoid clutter.

Note that  $\pi^s$  should not be confused with the marginal distribution on site  $s$  since it simply denotes the probability of a particular phase  $x_s$  at that site given specific neighbourhood configuration  $x_{N_s}$  (as specified by  $\pi^s(x)$  where  $x = x_s x_{N_s} x_{S \setminus (s \cup N_s)}$ ).

**Definition 2.2.9.** *Positivity Condition*

A probability distribution  $P$  satisfies the **positivity condition** if,  $\forall x \in \Lambda^S$  we have,

$$P(x) > 0$$

**Theorem 2.2.1.** *Uniqueness of Distribution for a Local Specification*

If two MRF's on a finite configuration space  $\Lambda^S$  have distributions that satisfy the positivity condition and have the same local specifications, then they are identical.

**Theorem 2.2.2.** *Hammersley-Clifford (Gibbs-Markov Equivalence)*

The distribution of a MRF satisfying the positivity condition is a Gibbs distribution (it can be specified by an energy derived from a Gibbs potential). Additionally, the local characteristics are given by,

$$\pi^s(x) = \frac{\exp\left(-\sum_{C \subset N_s} V_C(x)\right)}{\sum_{\lambda \in \Lambda} \exp\left(-\sum_{C \subset N_s} V_C(\lambda_s x_{S \setminus s})\right)}.$$

Note that  $s \in C$  is not necessarily true for all  $C \subset N_s$ . Since  $x_{N_s}$  is given, the potentials on cliques which do not contain  $s$  can be “factored” out of the top and bottom. Thus, alternatively, the sum  $\sum_{C \ni s}$  can be used in place of  $\sum_{C \subset N_s}$  to denote the sum over all  $C$  that contain  $s$ .

This is an important result since it gives an explicit form for the MRF. The proof

in the forward direction simply involves rearranging the potentials in the energy function in such a way that demonstrates the Markov property by cancellation of potentials outside some relevant neighbourhood. The converse part of the proof, due to Hammersley and Clifford 1971 [14], is based on the Möbius formula [13].

Thus far, we know that MRF's are Gibbs fields and that their distributions can be specified by Gibbs potentials. However, we have not addressed whether or not these Gibbs potentials are unique. As it turns out, they are not unique. In fact, this can be easily shown by adding some constant to every potential. The result is a new Gibbs potential with higher/lower overall energy, but the same associated Gibbs distribution. Fortunately, some uniqueness can be found with a special type of potentials called normalized potentials. The existence and uniqueness of Gibbs fields have been studied in [9, 10, 11].

**Definition 2.2.10.** *Normalized Potentials*

Let  $\lambda \in \Lambda$  be some phase and  $\{V_C\}_{C \subset S}$  be some Gibbs potential. If,

$$(x_t = \lambda, t \in C) \implies (V_C(x) = 0),$$

then we say that the Gibbs potential is **normalized with respect to**  $\lambda$ .

**Theorem 2.2.3.** *Uniqueness of Normalized Potential*

If a Gibbs distribution is specified by an energy derived from a Gibbs potential normalized with respect to some  $\lambda \in \Lambda$ , then there exists no other  $\lambda$ -normalized potential which specifies the same Gibbs distribution.

### 2.2.5 Ising Model

When modelling a class of images, we can use a random field where the nodes of the graph (or sites) represent the individual pixels of an image (or configuration). In this context, Gibbs fields can be especially useful when attempting to specify a model for images with particular local interactions. This is done by selecting the appropriate neighbourhood, cliques, and potential functions. An example can be found in [12] where MRFs were used in texture modelling.

A well known example of a MRF is the “Ising model” [15, 22]. This model was studied primarily for its usefulness in modelling the ferromagnetic interactions between iron atoms. Here, the phase space,  $\Lambda$ , is  $\{1, -1\}$ , where the two values account for the magnetic “spin” of the atom at some site. The set of sites is simply  $S = \mathbb{Z}_m^2$  (where  $\mathbb{Z}_m = \{0, 1, \dots, m-1\}$ ), that is some finite “surface” of sites arranged on a square integer enumerated grid. A commonly used neighbourhood is the 4-point neighbourhood composed of sites immediately above, below, left, and right of some site  $s$ . The cliques for this model are limited to just individual nodes, and sets of two neighbouring nodes. The potential functions on these cliques are defined to be,

$$V_{\{s\}}(x) = -\frac{\bar{H}}{k}x_s,$$

$$V_{\{s,t\}}(x) = -\frac{J}{k}x_sx_t.$$

Here,  $k$  is the Boltzmann constant,  $H$  is a value representing the biasing effect of

external magnetic fields, and  $J$  is a value representing the coupling effects of elementary magnetic dipoles. We make the following adjustments,

$$\theta_1 = -\frac{\bar{H}}{k},$$

$$\theta_2 = -\frac{J}{k},$$

where in this context,  $\theta_1$  and  $\theta_2$  can simply be viewed as governing parameters for site interactions; the former being restricted to a single node (the tendency for individual sites towards a given phase), and the latter for pairs of nodes (phase coupling effect for neighbouring sites). From this, the following energy, Gibbs distribution, and local specification can be found,<sup>3</sup>

$$\varepsilon(x) = \sum_{\{s\} \subset S} \theta_1 x_s + \sum_{\{s,t\} \subset S} \theta_2 x_s x_t,$$

$$\pi(x) = \frac{1}{Q(\theta)} \exp \left( - \sum_{\{s\} \subset S} \theta_1 x_s - \sum_{\{s,t\} \subset S} \theta_2 x_s x_t \right),$$

$$\pi^s(x) = \frac{\exp \left( -\theta_1 x_s - \sum_{\{s,t\} \subset N_s} \theta_2 x_s x_t \right)}{\sum_{\lambda \in \Lambda} \exp \left( -\theta_1 \lambda_s - \sum_{\{s,t\} \subset N_s} \theta_2 \lambda_s x_t \right)}.$$

$Q(\theta)$  shows that the partition function depends on the values of  $\theta_1$  and  $\theta_2$ . Historically, the Ising model typically used a 4-point neighbourhood where  $N_s$  includes the nodes just east, south, west, and north of site  $s$ . Figure 2.1 shows examples of

---

<sup>3</sup>Where potential functions are concerned,  $x_s x_t$  will usually refer to a mathematical operation between the phases of sites as opposed to denoting some given configuration on  $s$  and  $t$ .

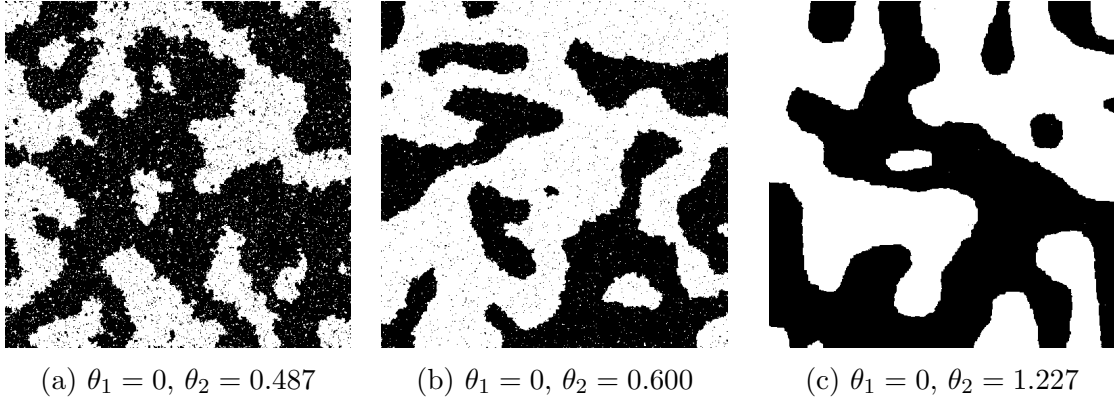


Figure 2.1: Sample Ising configurations with varying parameters

the Ising model with various parameters.

## 2.3 Homogenous and Nonhomogenous Markov Chains

The results discussed here will pertain specifically to Markov chains in discrete time with finite state space - this is all we need to proceed. We include short proofs, and omit others.

### 2.3.1 Overview of Basic Notions

- A **discrete-time stochastic process** is a sequence  $\{X_n\}_{n \geq 0}$  of random variables taking values on some state space  $\mathbf{X}$ .
- The stochastic process  $\{X_n\}_{N \geq 0}$  is called a **Markov chain** if  $\forall n \geq 0$ , and states  $i_0, \dots, i_{n-1}, i, j \in \mathbf{X}$ , we have that,

$$P(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, \dots, X_0 = i_0) = P(X_{n+1} = j | X_n = i).$$

- The **transition matrix** of a Markov chain at time  $n$  is the matrix  $\mathbf{P}_n = \{p_{n_{ij}}\}_{i,j \in \mathbf{X}}$ , where  $p_{n_{ij}} = P(X_{n+1} = j | X_n = i)$  is the one step probability of going from state  $i$  to state  $j$ . Note that  $\sum_{k \in \mathbf{X}} p_{n_{ik}} = 1$ .
- If  $\mathbf{P}_n$  is independent of  $n$ , then the Markov chain is called a **homogeneous Markov chain**.
- For homogeneous Markov chains, the  $n$  step transition matrix is given by,

$$\mathbf{P}_1 \cdots \mathbf{P}_n = \mathbf{P}^n.$$

- A distribution  $\mu$  on  $\mathbf{X}$  is called an **invariant distribution** for some homogeneous Markov process with transition matrix  $\mathbf{P}$  if it satisfies  $\mu\mathbf{P} = \mu$ . It is also often referred to as the **stationary distribution**.
- The **total variation** distance between two distributions  $\mu$  and  $\nu$  on  $\mathbf{X}$  is the  $L^1$ -norm,<sup>4</sup>

$$\|\mu - \nu\| = \sum_{k \in \mathbf{X}} |\mu_k - \nu_k|.$$

**Definition 2.3.1.** A Markov chain is **irreducible** if  $\forall i, j \in \mathbf{X}, \exists n$  such that  $\mathbf{P}_{i,j}^n > 0$ .

**Definition 2.3.2.** The **period**  $k$  of some state  $i$  is defined as,

$$k = \gcd\{n : \mathbf{P}_{i,i}^n > 0\}$$

A Markov chain is **aperiodic** if  $k = 1 \forall i \in \mathbf{X}$ .

---

<sup>4</sup>In general, when we refer to a distribution  $\mu$  on the finite state space  $\mathbf{X}$ , we think of it in terms of a vector where  $\mu_i$  represents the  $i$ th entry, i.e., the probability of the  $i$ th state.

**Definition 2.3.3.** The **support** of a distribution refers to the set of states on which it is strictly positive. Distributions with disjoint support are said to be **orthogonal**.

The following are some important results by Dobrushin [8].

**Definition 2.3.4.** *Dobrushin's Coefficient*

Let  $\mathbf{P}$  be the transition matrix for some Markov chain on a finite state space, then Dobrushin's coefficient is defined,

$$\delta(\mathbf{P}) = \frac{1}{2} \max_{i,j \in \mathbf{X}} \sum_{k \in \mathbf{X}} |p_{ik} - p_{jk}| = \frac{1}{2} \max_{i,j \in \mathbf{X}} \|p_{i\cdot} - p_{j\cdot}\|.$$

Note that  $0 \leq \delta(\mathbf{P}) \leq 1$  with it being equal to 1 if and only if the two distributions (rows of  $\mathbf{P}$ )  $p_{i\cdot}$  and  $p_{j\cdot}$  have disjoint supports and 0 if the rows of  $\mathbf{P}$  are identical. Thus, the coefficient can be thought of as a measure of orthogonality.

**Lemma 2.3.1.** Let  $\mu$  and  $\nu$ , and  $\mathbf{P}$  be probability distributions and transition matrix on  $\mathbf{X}$  respectively, then,

$$\|\mu\mathbf{P} - \nu\mathbf{P}\| \leq \delta(\mathbf{P})\|\mu - \nu\|.$$

**Theorem 2.3.1.** *Dobrushin's Inequality*

$$\delta(\mathbf{PQ}) \leq \delta(\mathbf{P})\delta(\mathbf{Q}).$$

## 2.3.2 Convergence of Homogeneous Markov Chains

We are now ready to present some important results regarding the convergence of homogeneous and inhomogeneous Markov chains. Again, see [3, 26] for more in-depth



treatment.

**Definition 2.3.5.** *Primitive Homogenous Markov Chain*

Let  $\mathbf{P}$  be the transition matrix of a Markov chain. If  $\exists \tau \in \mathbb{Z}^+$  such that  $\forall i, j \in \mathbf{X}$  we have,

$$\mathbf{P}_{i,j}^\tau > 0,$$

then the homogeneous Markov chain is said to be **primitive**.

**Theorem 2.3.2.** *A Markov chain is primitive if and only if it is irreducible and aperiodic.*

**Lemma 2.3.2.** *If  $\mathbf{P}$  is the transition matrix for a homogeneous Markov chain, then  $\{\delta(\mathbf{P}^n)\}_{n \geq 0}$  is a decreasing sequence. Specifically, if  $\mathbf{P}$  is primitive, then the sequence decreases to 0.*

**Theorem 2.3.3.** *Convergence of Homogeneous Markov Chains*

Let  $\mathbf{P}$  be the transition matrix for a primitive homogeneous Markov chain on a finite state space  $\mathbf{X}$  with invariant distribution  $\mu$ , then for all distributions  $\nu$ , uniformly we have,

$$\lim_{n \rightarrow \infty} \nu \mathbf{P}^n = \mu.$$

**Lemma 2.3.3.** *Reversible Markov Chain*

If for some distribution  $\mu$ , the transition matrix  $\mathbf{P}$  for a Markov chain satisfies,

$$\mu_i p_{ij} = \mu_j p_{ji}$$

$\forall i, j \in \mathbf{X}$ , called the **detailed balance equations**, then the chain is said to be **reversible** and  $\mu$  is an invariant distribution for  $\mathbf{P}$ .

*Proof.*

$$\mu_{p \cdot j} = \sum_{i \in \mathbf{X}} \mu_i p_{ij} = \sum_{i \in \mathbf{X}} \mu_j p_{ji} = \mu_j.$$

□

This result is often known as Dobrushin's Theorem [8]. The proof for this theorem will not be shown, but it is useful to note for some sequence of Markov transition matrices  $\{\mathbf{P}_n\}_{n \geq 0}$  and distributions  $\mu$  and  $\nu$ ,  $\delta(\mathbf{P}_0 \cdots \mathbf{P}_n) \rightarrow 0$  implies,

$$\|\mu \mathbf{P}_0 \cdots \mathbf{P}_n - \nu \mathbf{P}_0 \cdots \mathbf{P}_n\| \leq 2\delta(\mathbf{P}_0 \cdots \mathbf{P}_n) \rightarrow 0.$$

That is, some convergence or “loss of memory over time” takes place in an asymptotic manner.

### 2.3.3 Gibbs Sampler

Sampling from a MRF is problematic because the configuration space  $\Lambda^S$  is too large for direct sampling methods. In particular, as  $S$  increases in size, the partition function becomes computationally intractable. The Gibbs Sampler, coined by D. and S. Geman [12], avoids this problem by setting up a Markov chain  $\{X_n\}_{n \geq 0}$  where  $X_n$  takes on values from the finite configuration space  $\Lambda^S$ , and by which the invariant distribution is the Gibbs distribution associated with the MRF of interest.

To start, the transition probabilities between two configuration  $x, y \in \Lambda^S$  are needed. A natural approach is to take a look at the set of sites on which  $x$  and  $y$  hold different values (denote this set by  $A$ ), and using the Markov property to determine the probability of a configuration switch on the set  $A$  conditioned on relevant

neighbours. That is,

$$p_{xy} = P(X_{n+1} = x | X_n = y) = P(X_A = x_A | X_{S \setminus A} = x_{S \setminus A}),$$

where  $x_A \neq y_A$ ,  $x_{S \setminus A} = y_{S \setminus A}$ , and  $x = x_A x_{S \setminus A}$ ,  $y = y_A y_{S \setminus A} = y_A x_{S \setminus A}$ . The last part of the above expression bears resemblance to *local characteristics* - this gives rise to the following definition.

**Definition 2.3.6.** *Local Interactions*

The *local interactions*<sup>5</sup> of a MRF on set  $A \subset S$  are given by,

$$\pi^A(x) = P(X_A = x_A | X_{S \setminus A} = x_{S \setminus A}).$$

It is easy to see that local characteristics are simply a special case of local interactions when  $A \subset S$ ,  $A = \{s\}$ ,  $s \in S$ .

**Lemma 2.3.4.** *Let  $A \subset S$ . For a Gibbs field with distribution  $\pi$ , we have the following local interactions,*

$$\pi^A(x) = \frac{\exp(-\varepsilon(x_A x_{S \setminus A}))}{\sum_{y_A} \exp(-\varepsilon(y_A x_{S \setminus A}))}.$$

---

<sup>5</sup>This terminology is not commonly used in literature. The term *local characteristics* are sometimes used interchangeably between what's defined here as *local interactions* and *local characteristics*.

*Proof.*

$$\begin{aligned}
P(X_A = x_A | X_{S \setminus A} = x_{S \setminus A}) &= \frac{P(X = x_A x_{S \setminus A})}{P(X_{S \setminus A} = x_{S \setminus A})} = \frac{\pi(x_A x_{S \setminus A})}{\sum_{y_A} \pi(y_A x_{S \setminus A})} \\
&= \frac{\frac{1}{Q} \exp(-\varepsilon(x_A x_{S \setminus A}))}{\sum_{y_A} \frac{1}{Q} \exp(-\varepsilon(y_A x_{S \setminus A}))} \\
&= \frac{\exp(-\varepsilon(x_A x_{S \setminus A}))}{\sum_{y_A} \exp(-\varepsilon(y_A x_{S \setminus A}))}.
\end{aligned}$$

□

**Lemma 2.3.5.** *A Gibbs distribution and its associated local interactions satisfy the detailed balance equations. That is, let  $x, y \in \Lambda^S$  and the set  $A \subset S$  be the sites on which  $x$  and  $y$  differ in phase, then,*

$$\pi(x)p_{xy} = \pi(y)p_{yx}.$$

*Proof.* By Lemma 2.3.4

$$\begin{aligned}
\pi(x)p_{xy} &= \pi(x)\pi^A(y_A x_{S \setminus A}) = \frac{1}{Q} \exp(-\varepsilon(x_A x_{S \setminus A})) \frac{\exp(-\varepsilon(y_A x_{S \setminus A}))}{\sum_{z_A} \exp(-\varepsilon(z_A x_{S \setminus A}))} \\
&= \frac{1}{Q} \exp(-\varepsilon(y_A x_{S \setminus A})) \frac{\exp(-\varepsilon(x_A x_{S \setminus A}))}{\sum_{z_A} \exp(-\varepsilon(z_A x_{S \setminus A}))} \\
&= \pi(y)\pi^A(x_A x_{S \setminus A}) = \pi(y)p_{yx}.
\end{aligned}$$

Note that  $y = y_A y_{S \setminus A} = y_A x_{S \setminus A}$ .

□

By Lemma 2.3.3, this means that the Gibbs distribution  $\pi$  is an invariant distribution for a homogeneous Markov chain  $\{X_n\}_{n \geq 0}$ , where the transition matrix  $\mathbf{P}$  is defined by  $\{p_{xy}\}_{x,y \in \Lambda^S}$  where  $p_{xy}$  are the local interactions.

Furthermore, by Theorem 2.3.3, given any initial distribution  $\nu$  on  $\Lambda^S$ , we have that,

$$\lim_{n \rightarrow \infty} \nu \mathbf{P}^n = \pi.$$

This means that by starting with a sample from distribution  $\nu$ , a sample from  $\pi$  can be approximately attained by conducting a large number of probability experiments using  $\mathbf{P}$ .

Typically, experiments are done on individual sites using the easily computable local characteristics (i.e. sample  $x_{n+1}$  and sample  $x_n$  differ at a single site). Site alterations follow a specified **visiting scheme**, and once every site has been updated, it is said that a **sweep** has been completed. This idea of *sweeps* is rather important in ensuring that the Markov chain is primitive [26].<sup>6</sup>

It is also interesting to note that with some adjustments to the transition probabilities by introducing a temperature **schedule**, the Gibbs sampler can be adjusted for simulated annealing; a technique useful in finding maximal modes on some configuration space [12, 3, 26].

---

<sup>6</sup>That is, by making sure each site undergoes one “alteration”, we are demonstrating a  $\tau$  such that  $\forall i, j \in \mathbf{X}, \mathbf{P}_{i,j}^\tau > 0$ , hence showing that the Markov chain is primitive.

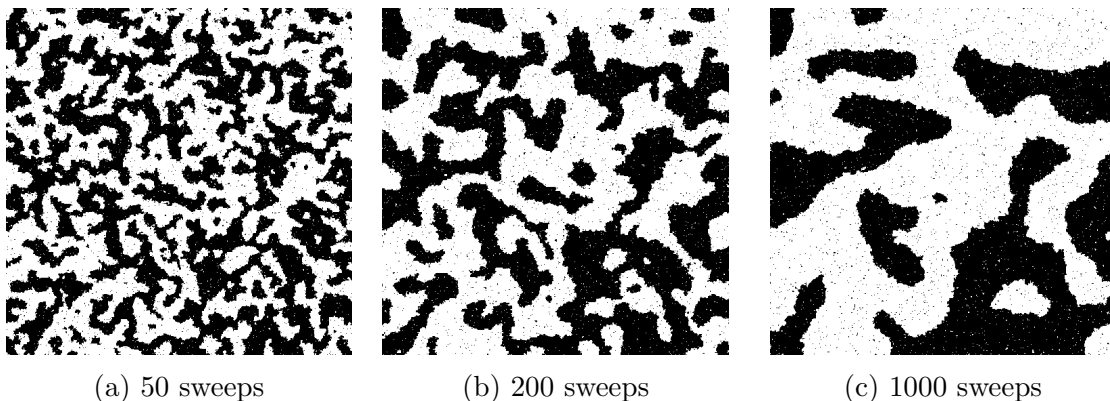


Figure 2.2: Generating an Ising sample from a noise seed ( $\theta_1 = 0$ ,  $\theta_2 = 0.600$ )

## 2.4 Information Theory and Source Coding

Some knowledge of information theory is assumed here. Since we are primarily interested in encoding to binary bits, all logarithms are base 2 unless otherwise specified.

### 2.4.1 Overview of Basic Information Theory

A **discrete source** is a discrete random variable taking on values in the **alphabet**  $\Lambda$ . Claude Shannon's **source coding theorem** in 1948 demonstrated that discrete sources could not be compressed below a certain threshold without loss of information [25]. This entity is known as the **entropy** of the source.

**Definition 2.4.1.** *Entropy for a Discrete Source*

For a discrete source with source alphabet  $\Lambda$  and probability mass function (pmf)  $p$ , the **entropy** of the source is defined as,

$$H(p) = - \sum_{x \in \Lambda} p(x) \log p(x) = E_p \left[ \log \frac{1}{p(X)} \right].$$

**Definition 2.4.2.** *Conditional Entropy*

Let  $X$  and  $Y$  be discrete sources with alphabets  $\Lambda_X$  and  $\Lambda_Y$ , and let  $p(x, y)$  be the joint distribution for  $(X, Y)$ . The **conditional entropy** of  $X$  condition on  $Y$  is given by,

$$\begin{aligned} H(X|Y) &= - \sum_{x \in \Lambda_X} \sum_{y \in \Lambda_Y} p(x, y) \log \frac{p(x, y)}{p(y)} \\ &= - \sum_{x \in \Lambda_X} \sum_{y \in \Lambda_Y} p(x, y) \log p(x|y). \end{aligned}$$

For stochastic processes, the entropy rate is defined as follows [25],

**Definition 2.4.3.** *Entropy Rate*

The **entropy rate** and **conditional entropy rate** of a stochastic process  $\mathcal{X} = \{X_N\}_{n \geq 0}$  are respectively defined,

$$\begin{aligned} H(\mathcal{X}) &= \lim_{n \rightarrow \infty} \frac{1}{n} H(X_0, \dots, X_n), \\ H'(\mathcal{X}) &= \lim_{n \rightarrow \infty} H(X_n | X_{n-1}, \dots, X_0), \end{aligned}$$

whenever the limit exists.

For a stationary process, the limits that define the entropy rate  $H(\mathcal{X})$  and the conditional entropy rate  $H'(\mathcal{X})$  exist and are equal. Moreover, if a Markov chain taking values on  $\Lambda$  with transition matrix  $\mathbf{P}$  has an invariant distribution  $\mu$ , then by letting  $X_0 \sim \mu$ , we have, [4]

$$H(\mathcal{X}) = - \sum_{i \in \Lambda} \sum_{j \in \Lambda} \mu_i p_{ij} \log p_{ij} = H(X_1 | X_0).$$

The relative entropy, also called the Kullback-Leibler divergence, is a well known

measurement of dissimilarity between two distributions [16].

**Definition 2.4.4.** *Kullback-Leibler Divergence*

Let  $p$  and  $q$  be pmfs over the same alphabet  $\Lambda$ . The **Kullback-Leibler divergence** between  $p$  and  $q$  is defined to be,

$$D(p||q) = \sum_{x \in \Lambda} p(x) \log \frac{p(x)}{q(x)}.$$

## 2.4.2 Lossless Source Coding

**Definition 2.4.5.** *Codes*

- A source **code** is a function  $C : \Lambda \mapsto D^*$ , where  $D^*$  is the set of finite-length string of symbols from a  $D$ -ary alphabet. The code is said to be **nonsingular** if  $C$  is injective.
- The **extension**  $C^*$  of a code  $C$  is the mapping from finite-length strings of  $\Lambda$  to that of  $D$ , i.e.  $C(x_1 \dots x_n) = C(x_1) \dots C(x_n)$ .
- The length of the **codeword**  $C(x)$  associated with source symbol  $x$  is denoted  $l(x)$ . The **expected length** of the code,  $L(C)$  is,

$$L(C) = \sum_{x \in \Lambda} p(x)l(x) = E[l(x)],$$

if  $X \sim p$ .

- A code  $C$  is **uniquely decodable** if its extension is nonsingular. The code is also **instantaneous** or a **prefix code** if no codeword is a “prefix” of another; the code self-punctuates and the end of a codeword is immediately recognizable.



In particular, a rather important result regarding uniquely decodable codes is given by [17],

**Theorem 2.4.1.** *Kraft-McMillan*

*Given any uniquely decodable  $D$ -ary code for a discrete source on  $\Lambda$ , we have,*

$$\sum_{x \in \Lambda} D^{-l(x)} \leq 1.$$

*Also, conversely, for any set of codeword lengths that satisfy the above inequality, it is possible to construct a uniquely decodable code with these code lengths.*

The coding process involves an **encoder** that uses a code  $C$  to map source symbols from  $\Lambda$  to  $D$ -ary codewords. A **decoder** is then used to transform the  $D$ -ary codewords back to source symbols from  $\Lambda$ .

In particular, a distinction is made between **lossless coding** and **lossy coding**; the former requires the perfect reconstruction of source symbols while the latter does not. In either case, the goal is usually to minimize the expected code length for efficient storage purposes.

**Definition 2.4.6.** *Optimal code*

*A code  $C^*$  is optimal if  $L(C^*) \leq L(C)$  for all other codes  $C$ .*

Other factors such as robustness to error and minimization of distortion also come into play. The scope of this thesis covers only the lossless compression of MRFs, with the main focus being that of data compression - i.e. small  $L(C)$ . From this point forth, we restrict ourselves to binary codes - that is,  $D = 2$ . We now go over some well known results regarding optimal codes [25].

**Theorem 2.4.2.** *Bounds for the Expected Length of an Optimal Code*

Let  $X$  be a discrete source on a finite alphabet  $\Lambda$  of with probability mass function (pmf)  $p$ , and let  $C^*$  be an optimal code for this source. Then,

$$H(p) \leq L(C^*) < H(p) + 1.$$

Instead of encoding symbols from  $\Lambda$ , it is often useful to encode symbols from  $\Lambda^n$ ,  $n \in \mathbb{Z}^+$  instead, i.e., encode strings of length  $n$  at a time. With a minor adjustment to notation, for a code  $C_n : \Lambda^n \mapsto 2^*$ , we define the expected codeword length per symbol by,

$$L_n(C_n) = \frac{1}{n} \sum_{x^n \in \Lambda^n} p(x^n) l(x^n),$$

where  $x^n = (x_1 \dots x_n)$ . The following establish some bounds for block coding and stationary processes [25, 4].

**Theorem 2.4.3.** *Let  $X_1, \dots, X_n$  be discrete random variables taking values on a finite alphabet  $\Lambda$ . Let  $C_n^*$  be an optimal code for this source of super-symbols from  $\Lambda^n$ , then*

$$\frac{H(X_1, \dots, X_n)}{n} \leq L_n(C_n^*) < \frac{H(X_1, \dots, X_n)}{n} + \frac{1}{n}.$$

Additionally, if  $\mathcal{X} = \{X_N\}_{n \geq 0}$  is a stationary process, then the minimum expected codeword length per symbol approaches the entropy rate,

$$\lim_{n \rightarrow \infty} L_n(C_n^*) = H(\mathcal{X}).$$

### 2.4.3 Shannon-Fano-Elias Coding

The Shannon-Fano-Elias code [4] is a code that maps each source symbol to a given subinterval of  $[0, 1]$ . This is done by using the cumulative distribution function of the source to allot the subintervals, or “bins”. The codeword for each bin is then determined by selecting an appropriate number from the bin so that altogether the codeword lengths meet the uniquely decodable criteria.

Given the cumulative distribution function for some discrete source  $X$  taking values on some ordered set,

$$F(x) = \sum_{y \leq x} p(y),$$

define the following,

$$\bar{F}(x) = \sum_{y < x} p(y) + \frac{1}{2}p(x).$$

That is,  $\bar{F}(x)$  is the midpoint between  $F(x)$  and  $F(x^*)$  where  $x^*$  is the symbol immediately smaller than  $x$ . If all the source symbols occur with non-zero probability, then for  $a \neq b$  ( $a, b \in \Lambda$ ), we have  $\bar{F}(a) \neq \bar{F}(b)$ .

$\bar{F}(x)$  itself is not a good choice for a codeword since it is a real number and will require an infinite number of bits, in general, to encode. Thus instead  $\bar{F}(x)$  is approximated by using only the first  $l(x)$  bits where  $l(x) = \lceil \log p(x)^{-1} \rceil + 1$ . Denote the truncated  $\bar{F}(x)$  by  $\lfloor \bar{F}(x) \rfloor_{l(x)}$ , observe that,

$$\sum_{x \in \Lambda} D^{-l(x)} \leq 1$$

and,

$$\bar{F}(x) - \lfloor \bar{F}(x) \rfloor_{l(x)} < \frac{1}{2^{l(x)}} < \frac{1}{2p(x)^{-1}} = \frac{p(x)}{2}.$$

The first expression shows that  $l$  satisfy the inequality of Theorem 2.4.1, and the second shows that  $\lfloor \bar{F}(x) \rfloor_{l(x)}$  retains enough precision to stay within the appropriate subinterval. Additionally, it is not hard to show that the code is instantaneous. The expected length of the code is,

$$L(C) = \sum_{x \in \Lambda} p(x) \left( \left\lceil \log \frac{1}{p(x)} \right\rceil + 1 \right) < H(X) + 2.$$

An extension of Shannon-Fano-Elias coding is arithmetic coding.

#### 2.4.4 Universal and Arithmetic Coding

Many codes rely on the assumption that the source distribution is known. In practice, this typically is not the case. Usually the only assumptions made about the source is that it belongs to a certain fixed family of source. Loosely speaking, a sequence of codes  $\{C_n\}_{n \geq 0}$  that compress each source in some family of sources asymptotically to its entropy rate is called a **universal** coding scheme [4].

Another problem that arises is the integer constraint on length of codewords. As seen from Theorem 2.4.2, even for optimal codes there is potential for up to 1 bit per symbol worth of inefficiency. By Theorem 2.4.3, this loss can be reduced by using block coding on strings of source symbols from  $\Lambda^n$ , but this will require the decoder to wait for  $n$  symbols before any decoding can be done. Thus, for real-time applications, block coding (compressing efficiently with large  $n$ ) can be impractical. Also, the computational complexity of determining a code (e.g. Huffman code) for super-symbols grows exponentially as  $n$  gets large, meaning that a lot of time may

need to be devoted to pre-computations, rendering the technique useless for any ad-hoc situations. Arithmetic coding offers a solution which essentially allows for block coding on-the-go with significantly less computational complexity [4].

Using the approach from Shannon-Fano-Elias coding, arithmetic coding uses subintervals of the unit interval to represent source symbols. It uses incoming source symbols to empirically construct a code that gets incrementally better in the block coding sense; as more symbols arrive, the subintervals become “finer”.

Define,

$$\hat{F}(x^n) = \sum_{y^n < x^n} p(y^n).$$

That is,  $\hat{F}(x^n)$  is the value of the cumulative distribution function for the source symbol sequence immediately “smaller” than  $x^n$ . Note that when  $n = 1$ , we can relate this directly to the Shannon-Fano-Elias code by,

$$\frac{F(x) + \hat{F}(x)}{2} = \bar{F}(x).$$

It is easy to show that,

$$\hat{F}(x^n) = \hat{F}(x^{n-1}) + p(x^{n-1}) \sum_{y_k < x_k} p(y_k | x^{k-1}).$$

This means that the subintervals at each stage can be computed using values from the previous stage. If the  $p(y_k | x^{k-1})$ 's are easy to compute, then the interval  $[\hat{F}(x^n), F(x^n)]$  (used to represent  $x^n$ ) can be easily computed as well. Shannon-Fano-Elias coding then tells us that such a coding scheme yields an expected code length

of,

$$L(C^n) < H(X^n) + 2.$$

Where  $H(X^n) = H(X_0, \dots, X_n)$ , and so it is not hard to see that the expected length per symbol  $L_n(C^n)$  approaches entropy.

When compared to Shannon-Fano-Elias coding, one differentiating feature of arithmetic coding is that no specific codewords are assigned for each interval, rather a subinterval of the current subinterval is chosen at the next stage, hence only an explicit expression of  $\hat{F}(X)$  is needed for the actual code. This also means that source symbols can be encoded “on-the-go” and be decoded in the same manner since they simply amount to some subinterval search.

The main challenge of arithmetic coding is the fact that the interval sizes decrease rapidly as  $n$  increases and so requires high floating point precision. Also, even though the source can be encoded in real time, the number of symbols it takes before the next subinterval is found is not constant. Nonetheless, there exist practical algorithms which solve these problems [24, 20].

For the purpose of this thesis, all we need to know is that it is possible to approach the entropy asymptotically via arithmetic coding. In particular note that the  $p(y_k|x^{k-1})$ 's are easily computable for Markov sources.

## 2.5 Estimation and Inference

Where MRF estimation and inference are concerned, past work has focused primarily on MRFs with only simple cliques. This was a natural choice since simple cliques translate nicely to the notion of nodes and edges in graphs, thus allowing various

tools from graph theory and computer science to be utilized. Also, most of the clique potentials studied in the past have largely been of the “Ising type”<sup>7</sup> since Ising’s model played a motivating role in the study of MRFs. For these reasons, the methods described here will generally apply specifically to MRFs with simple cliques and “Ising” interactions. In Chapter 4, we generalize some these methods for use on MRFs which contain complex cliques. Also, we will defer the proofs to Chapter 4.

### 2.5.1 Statistics and Moments

One particular class of MRFs<sup>8</sup> can also be specified using a parameter  $\theta \in \Theta$ , and a collection of statistics  $t$ . For example, in the 4-point neighbourhood Ising case, the probability for a configuration  $x \in \Lambda^S$  is given by,

$$\pi(x) = \frac{1}{Q(\theta)} \exp \left( - \sum_{\{s\} \subset S} \theta_1 x_s - \sum_{\{s,t\} \subset S} \theta_2 x_s x_t \right).$$

This could be expressed alternatively as,

$$\pi(x) = \frac{1}{Q(\theta)} \exp (-\theta_1 t_1(x) - \theta_2 t_2(x)) = \exp (-\langle \theta, t \rangle - \ln Q(\theta)),$$

---

<sup>7</sup>Self-potential and pair-wise interactions.

<sup>8</sup>By this we mean MRFs which are specified by a special class of potentials. We discuss this further in Chapter 4 when we talk about linear parameter potentials.

where  $t_1(x)$ , and  $t_2(x)$  are the statistics,

$$t_1(x) = \sum_{\{s\} \subset S} x_s,$$

$$t_2(x) = \sum_{\{s,t\} \subset S} x_s x_t,$$

and,  $\theta$  and  $t$  are the vectors containing the respective  $\theta$ 's and statistics. Note that for a given configuration  $x \in \Lambda^S$ , the vector  $t$  does not show dependency on  $x$ . Also, the partition function  $Q(\theta)$  has dependency on  $t$ , but we leave this out. For this thesis, the MRFs we discuss will generally share common statistics  $t$ , but vary in  $\theta$ .

In this way, the set of possible  $\theta \in \Theta$  form a coordinate system for MRFs of a given statistic  $t$ . Also, for a given  $\theta$ ,  $\mu$  denotes the vector containing the expected value of statistic  $t$  under the MRF specified by  $\theta$  - we call this the **moment** parameter of the MRF. If  $t$  is minimal, then there is a one-to-one correspondence between a given  $\theta$  and some moment  $\mu$  [2]<sup>9</sup>. Thus, for MRFs with minimal statistic  $t$ , it is also possible to specify the MRF using the vectors  $\mu$  and  $t$ .

The statistic  $t$  is said to be **positively correlated** if for some  $\theta$  with positive entries, the covariance (on the MRF) between any two components of  $t$  is greater or equal to zero. If  $t$  is minimal, then these covariances are strictly positive.

Finally, concerning notation, for  $L \subset V$  on graph  $G = (V, E)$ , we use  $\theta_L$ ,  $\mu_L$  and  $t_L$  to denote the vectors that contain the entries of  $\theta$ ,  $\mu$ , and  $t$  which involve only nodes from  $L$ .

---

<sup>9</sup> $t$  is minimal if its components are affinely independent.



### 2.5.2 Least Squares Estimation

For a Gibbs field, it can be easily shown that,

$$\ln \left( \frac{P(X_s = \lambda, X_{N_s} = x_{N_s})}{P(X_s = \lambda^*, X_{N_s} = x_{N_s})} \right) = - \sum_{C \subset N_s} \left( V_C(\lambda_s x_{S \setminus s}) - V_C(\lambda_s^* x_{S \setminus s}) \right).$$

In the case of the Ising model, we have for  $\lambda, \lambda^* \in \Lambda$ ,

$$\ln \left( \frac{P(X_s = \lambda, X_{N_s} = x_{N_s})}{P(X_s = \lambda^*, X_{N_s} = x_{N_s})} \right) = -\theta_1(\lambda - \lambda^*) - \theta_2 \sum_{\{s,t\} \subset N_s} (\lambda - \lambda^*) x_t.$$

Note that  $P(X_s = \lambda, X_{N_s} = x_{N_s})$  and  $P(X_s = \lambda^*, X_{N_s} = x_{N_s})$  can be estimated empirically and thus the above is just a linear equation involving the unknowns  $\theta_1$  and  $\theta_2$ . The estimation procedure by [7] is as follows,

1. Collect data for various neighbourhood and site configurations.
2. Construct an over-determined system of equations with  $\theta_1$  and  $\theta_2$  unknowns (use the data collected in (1) to compute the relevant coefficients).
3. Solve by least squares method.

It is worth mentioning that the least squares (LS) method is the most easily computable method for estimating an MRF. This comes from the fact that it avoid computing the partition function and dealing with non-linear equations. The size of the system of equations increases exponentially with the number of sites  $|S|$ , and the size of phase space  $|\Lambda|$ .

### 2.5.3 Maximum Likelihood Estimation

The maximum likelihood estimate seeks to maximize the probability of some observations given some parameter, that is,

$$\hat{\theta} = \arg \max_{\theta \in \Theta} L(\theta),$$

where  $L(\theta)$  is the **likelihood** and  $\theta \in \Theta$  is the parameter of interest. Often, we will instead equivalently maximize the **log likelihood**  $\ln L(\theta)$  instead. Assuming that we make an estimate from  $n$  independent observations, the likelihood is written,

$$L(\theta) = \prod_{i=1}^n f(x_i|\theta),$$

where  $f$  is the relevant distribution for which we are estimating  $\theta$  - in our case, a Gibbs distribution. Note that  $x_i \in \Lambda^S \forall i$ . In the case of the Ising model,  $\theta = \{\theta_1, \theta_2\}$ , we have,

$$\hat{\theta} = \arg \max_{\theta \in \Theta} \left( -n \ln Q(\theta) - \theta_1 \sum_{i=1}^n x_{i_s} - \theta_2 \sum_{i=1}^n \sum_{(s,t) \in E} x_{i_s} x_{i_t} \right).$$

However since the partition function  $Q(\theta)$  is computationally intractable<sup>10</sup>,  $\hat{\theta}$  cannot be determined directly. Instead, an approximation can be attained via a local search given some starting parameter  $\theta$ . This is done by gradient ascent on the function  $\nabla Q(\theta) : \theta \mapsto L(\theta)$ . The gradient is given by,

$$\left\{ \frac{\partial}{\partial \theta_k} \ln (L(\theta)) \right\}.$$

---

<sup>10</sup>Recall that the partition function is a normalizing factor composed of the sum of energies across all possible configurations on  $\Lambda^S$ .

As it turns out, for the Ising case, we have,

$$\begin{aligned}\frac{\partial}{\partial \theta_1} \ln(L(\theta)) &= n \left( \mu(t_1) - t_1(x_1, \dots, x_n) \right), \\ \frac{\partial}{\partial \theta_2} \ln(L(\theta)) &= n \left( \mu(t_2) - t_2(x_1, \dots, x_n) \right),\end{aligned}$$

where  $t_1, t_2$  are the statistics given by,

$$\begin{aligned}t_1(x_1, \dots, x_n) &= \frac{1}{n} \sum_{i=1}^n \sum_{s \in S} x_{i_s}, \\ t_2(x_1, \dots, x_n) &= \frac{1}{n} \sum_{i=1}^n \sum_{(s,t) \in E} x_{i_s} x_{i_t}.\end{aligned}$$

and  $\mu(t_1), \mu(t_2)$  are their respective expected values on  $f_\theta$ .

Thus, minimizing the gradient amounts to a sort of moment matching to a sufficient training set. The estimation procedure [12] can be summed up as follows,

1. Using a training set of  $n$  samples, compute  $t_1(x_1, \dots, x_n)$  and  $t_2(x_1, \dots, x_n)$ .
2. Starting with a “guess” for  $\theta$ , approximate  $\mu(t_1)$ , and  $\mu(t_2)$  by generating sufficient samples from  $f_\theta$  using the Gibbs sampler, and computing their statistics.
3. Increment  $\theta$  by gradient ascent. Specifically,

$$\theta_{n+1} = \theta_n + \alpha \nabla Q(\theta),$$

where  $\alpha > 0$  is the step size.

4. repeat steps (2) and (3) until the gradient becomes small.

Finally, due to the fact that the Gibbs sampler has to be used at every stage of the iterative search, the maximum likelihood (ML) method is generally computationally time consuming. The number of samples needed for approximation increases exponentially with the number of sites  $|S|$ , and the size of phase space  $|\Lambda|$ .<sup>11</sup> The time to generate a sample increases with  $|S|$  and  $|\Lambda|$  as well.<sup>12</sup>

### 2.5.4 Belief Propagation

The Belief Propagation (BP) algorithm allows us to compute conditional probabilities and marginal probabilities on subsets of a graph  $G = (V, E)$ . For this thesis, this method becomes particularly useful in evaluating the relevant probabilities when conducting arithmetic coding [21, 23].

**Definition 2.5.1.** *Belief*

Given graph  $G = (V, E)$ , the **belief** for some set  $A \subset V$  is defined as  $Z_A = \{Z_A(x_A) : x_A \in \Lambda^A\}$  where,

$$Z_A(x_A) = \sum_{x_{V \setminus A}} \prod_{C \subset V} \exp(-V_C(x_A x_{V \setminus A})).$$

That is, a sum of configuration energies<sup>13</sup> over sites not in  $A$  for given  $x_A$ . Since the potentials determine the probability associated with a configuration, this summing process is analogous to computing the marginal on  $A$  and it is no surprise that  $Z_A(x_A) \propto p_A(x_A)$ .<sup>14</sup>

**Definition 2.5.2.** *Messages*

---

<sup>11</sup>Since the configuration space  $\Lambda^S$  grows exponentially

<sup>12</sup>The time it takes for a sweeps grows in proportion to  $\Lambda^S$

<sup>13</sup>We abuse the terminology here a bit since energy usually refers to the sum of potentials in a configuration.

<sup>14</sup>Note that  $Q(\theta) = \sum_{x_A} Z_A(x_A)$

Let  $G = (V, E)$  be some graph. For some cut-edge  $(i, j) \in E$ , the **message** from node  $j$  to node  $i$  is defined to be  $m_{j \rightarrow i} = \{m_{j \rightarrow i}(x_i) : x_i \in \Lambda\}$  where,

$$m_{j \rightarrow i}(x_i) = \sum_{x_j} \phi_{i,j}(x_i, x_j) Z_{j \setminus i}(x_j),$$

where  $\phi_{i,j}(x_i, x_j)$  denotes the **edge potential** on the edge  $(i, j)$  and  $Z_{j \setminus i}$  denotes the belief of node  $j$  on the subgraph  $G_{j \setminus i}$ .

If  $G$  is acyclic, then every edge in the graph is a cut-edge and we have the following theorems:

**Theorem 2.5.1.** *Belief Decomposition*

Let  $G = (V, E)$  be acyclic, then we have that  $\forall i \in V, x_i \in \Lambda$ ,

$$Z_i(x_i) = \varphi_i(x_i) \prod_{j \in \partial i} m_{j \rightarrow i}(x_i),$$

where  $\varphi_i(x_i)$  is the **self potential** on node  $i$ .

**Theorem 2.5.2.** *Message Recursion*

Let  $G = (V, E)$  be acyclic, then messages can be calculated recursively by,

$$m_{j \rightarrow i}(x_i) = \sum_{x_j} \phi_{i,j}(x_i, x_j) \varphi_j(x_j) \prod_{k \in \partial j \setminus i} m_{k \rightarrow j}(x_j),$$

where  $k \in \partial j \setminus i$  are the neighbours of  $j$  except  $i$ .

Note in the case of Ising model, the edge and self potentials are,

$$\begin{aligned}\varphi_i(x_i) &= \exp(-\theta_1 x_i), \\ \phi_{i,j}(x_i, x_j) &= \exp(-\theta_2 x_i x_j).\end{aligned}$$

The procedure for computing the belief for some node  $i$  involves treating node  $i$  as the root of a tree and recursively computing the messages from children nodes until leaf nodes are reached. Note that at each step a message is not passed back up to a parent node until a child node has received all relevant messages from its own children.

### 2.5.5 Clustering

For cyclic graphs, the BP algorithm can still be made to work by grouping nodes into **clusters** or **supernodes** in such a way that the resulting groups of nodes form an acyclic graph.

The potentials associated with supernode  $C_i$  and **superedge**  $(C_i, C_j)$  are defined as follows,

$$\begin{aligned}\varphi_{C_i}(x_{C_i}) &= \prod_{j \in C_i} \varphi_j(x_j) \prod_{i,j \in C_i} \phi_{i,j}(x_i, x_j), \\ \phi_{C_i, C_j}(x_{C_i}, x_{C_j}) &= \prod_{m \in C_i, n \in C_j} \phi_{m,n}(x_m, x_n).\end{aligned}$$

BP then proceeds in the “usual” sense. We discuss this in depth and in greater generalization (non-Ising) in Chapter 4.

## 2.6 Coding of MRF's

Coding of MRF's has been largely unexplored. [23] proposes two coding methods for binary images treated as MRF's on 2-D graphs. Both methods reduce computational complexity by appropriately choosing certain subsets that segment the image into disconnected components, or subgraphs, of fewer sites - the subsets chosen are referred to as **cutsets**.

The first is a lossy method whereby the cutsets are simply a one pixel boundary of subgraphs. These cutsets are encoded using standard procedures such as run length coding and no information is stored regarding the subgraphs. In the decoding process, the subgraphs are MAP reconstructed where the probabilities involved are governed by a set of of loop-filling theorems. We do not deal with this method in this thesis.

The second is a lossless method whereby the cutsets and subgraphs are encoded by assuming a MRF structure. The encoding process involves inference and arithmetic coding via the BP algorithm. This method is called **Reduced Cutset Coding**. This thesis adopts a similar method.

### 2.6.1 Entropy of MRFs

**Theorem 2.6.1.** *Entropy of MRF*

*For MRFs introduced in Section 2.5.1, <sup>15</sup> the entropy of the MRF can be expressed by,*

$$H(X; \theta) = \mathcal{Q}(\theta) - \nabla \mathcal{Q}(\theta)^T \theta,$$

*where  $\mathcal{Q}(\theta) \doteq \log Q(\theta)$ .*

---

<sup>15</sup>Specifiable by  $\theta$  and  $t$ .

This can be verified by direct computation. We use the notation  $H(X; \theta)$  to indicate the dependency on parameter  $\theta$ . Though not reflected by the notation, it should also be mentioned that there is dependency on  $G$  - that is, since a MRF distribution arises from potentials over cliques on the set of sites, the number of sites and their arrangement is significant to the entropy of the MRF.

### 2.6.2 Reduced MRFs

For an MRF on graph  $G$ , we will often denote the distribution of the MRF by  $p_G$  to clearly identify the graph (specifically we are interested in knowing the set of relevant sites). For some subset  $L \subset S$ , we denote the marginal distribution of  $X_L$  by,

$$p_G^L(x_L) = \sum_{x_{S \setminus L}} p_G(x_L x_{S \setminus L}).$$

A **reduced** MRF on  $G_L$ ,  $L \subset S$ , is a MRF specified by some  $\theta_L$ , and  $t_L$ . Its distribution is denoted by  $p_{G_L}$  and the probability of a given configuration  $x_L$  can be written as,

$$p_{G_L}(x_L) = \exp\{-\langle \hat{\theta}_L, t_L \rangle - \ln Q_L(\hat{\theta}_L)\}.$$

We use  $Q_L$  to indicate that the partition function is not the same as that of  $p_G$ . As mentioned before, in general, we are interested in MRFs which share the same statistic  $t$ , but differ in  $\theta$ ,<sup>16</sup> and so we drop notation which show dependency on  $t$ .

Lastly, we use the following notation to identify different MRF distributions,

1.  $p_{G, \theta}$ , is a MRF on graph  $G$  with parameter  $\theta$ .

---

<sup>16</sup>In the case of reduced MRFs, we are interested in MRFs on  $G_L$  that have the same statistic  $t$  (i.e.  $t_L$ ), but do not necessarily the same parameter  $\theta$  (i.e.  $\theta_L^* \neq \theta_L$ ).



2.  $p_{G,\theta}^L$ , is the marginal distribution on  $L \in S$  for an MRF on graph  $G$  with parameter  $\theta$ .
3.  $p_{G_L,\hat{\theta}_L}$ , is a reduced MRF on graph  $G_L$  with parameter  $\hat{\theta}_L$ .
4.  $p_{G_L,\mu_L}$ , is a reduced MRF on graph  $G_L$  with moments  $\mu_L$ .

For (4), note that if  $t$  is not minimal, then a  $\hat{\theta}_L$  giving giving rise to the distribution need not be unique. In this case, simply refers to some MRF with the moments  $\mu_L$  and do not care about the parameter. If the reduced MRF with distribution  $p_{G_L,\mu_L}$  has the same moments on  $L$  as some MRF on graph  $G$  with moments  $\mu$  and distribution  $p_{G,\theta}$ , then we call it a **moment-matching** reduced MRF for the MRF with distribution  $p_{G,\theta}$ .

**Definition 2.6.1.** *Statistical Manifolds*

- The set of all possible MRFs on graph  $G$  with statistic  $t$  is denoted by  $\mathcal{F}$ .
- The **e-flat submanifold** of some  $L \subset S$  is the set of MRFs for which the entries of parameter  $\theta$  not in  $\theta_L$  are set to zero. That is,

$$\mathcal{F}'_L(0) = \{p' \in \mathcal{F} : \theta \setminus \theta_L = 0\}.$$
<sup>17</sup>

- Given an MRF with distribution  $p_{G,\theta}$  and moments  $\mu$ , the **m-flat submanifold** of some  $L \subset V$  is the set of MRFs that share the same moments on  $\mu_L$ . That is,

$$\mathcal{F}''_L(\mu_L) = \{p'' \in \mathcal{F} : \mu''_L = \mu_L\}.$$

---

<sup>17</sup>We abuse the vector notation of  $\theta$  and  $\theta_L$  a bit. Also, note that we simply use the distribution to represent the MRF.

$\mathcal{F}'_L$  and  $\mathcal{F}''_L$  [2] are known as **orthogonal submanifolds** since the two partition  $\mathcal{F}$  [1]. The two intersect uniquely at an MRF  $p_{G,\theta^*}$ . The following Lemma states a well known result from information geometry [1].

**Lemma 2.6.1.** *Pythagorean Decomposition*

Given an MRF with distribution  $p_{G,\theta}$  with moments  $\mu$ . Take some subset  $L \subset S$ . Let  $\theta'$  be the parameter for some MRF in  $\mathcal{F}'_L(0)$ . Let  $\theta^*$  be the parameter representing the MRF at the intersection of  $\mathcal{F}'_L(0)$  and  $\mathcal{F}''_L(\mu_L)$ , then we have that,

$$D(p_{G,\theta} || p_{G,\theta'}) = D(p_{G,\theta} || p_{G,\theta^*}) + D(p_{G,\theta^*} || p_{G,\theta'}).$$

Reyes, in [23] shows further that this decomposition can be extended to reduced MRFs.

**Theorem 2.6.2.** *Pythagorean Decomposition for Reduced MRFs*

Given an MRF with distribution  $p_{G,\theta}$  with moments  $\mu$ . Take some subset  $L \subset S$  non-empty. Let  $\theta'$  be the parameter for some MRF in  $\mathcal{F}'_L(0)$ , then we have the following decomposition for the marginal distribution  $p_{G,\theta}^L$  on  $L$ , and the reduced MRF distribution  $p_{G_L,\theta'_L}$ ,

$$D(p_{G,\theta}^L || p_{G_L,\theta'_L}) = D(p_{G,\theta}^L || p_{G_L,\mu_L}) + D(p_{G_L,\mu_L} || p_{G_L,\theta'_L}),$$

where  $p_{G_L,\mu_L}$  is the unique moment-matching reduced MRF given by the intersection of the submanifolds.

Furthermore, [23] shows that,

**Theorem 2.6.3.** *Given an MRF with minimal and positive correlated statistic  $t$ , parameter  $\theta$ , and moments  $\mu$ , for a subset  $L \subset S$ , we have,*

$$H_G^L(X_L; \theta) \leq H_{G_L}(X_L; \mu_L) \leq H_{G_L}(X_L; \theta_L)$$

### 2.6.3 Reduced Cutset Coding (RCC)

As previously discussed, arithmetic coding allows us to encode on-the-go with expected code length which approaches the entropy asymptotically. At each given stage  $k$ , if  $p(y_k|x^{k-1})$  is easy to compute, then the coding distribution can be updated accordingly. Given some MRF, if we cluster the nodes accordingly to form an acyclic cluster graph of clusters  $\{A_i\}$ , then as discussed previously, we can compute the beliefs for each of these clusters using the BP algorithm. If we encode the clusters in some sequential manner, this means the probabilities at each stage can be calculated easily. In particular, note that,

$$\begin{aligned} P(X_{A_k} = x_{A_k} | x_{A_{k-1}}, \dots, x_{A_1}) &= \frac{P(X_{A_k} = x_{A_k}, X_{A_{k-1}} = x_{A_{k-1}}, \dots, X_{A_1} = x_{A_1})}{P(X_{A_{k-1}} = x_{A_{k-1}}, \dots, X_{A_1} = x_{A_1})} \\ &= \frac{Z_{(A_1 \cup \dots \cup A_k)}(x_{A_1} \dots x_{A_k})}{Z_{(A_1 \cup \dots \cup A_{k-1})}(x_{A_1} \dots x_{A_{k-1}})}, \end{aligned}$$

and so the computation amounts to a belief calculation via message collection at each stage.

For MRFs with large number of sites, the number of possible super symbols grows quite large to make computation difficult. [23] proposes that a cutset  $L \subset S$  be chosen to separate the MRF into disconnected components. Conditioning on the cutset  $L$ , these components can be encoded at a rate close to the conditional entropy

$H_G(X_{S \setminus L} | X_L)$ . As for the cutset  $L$ , it can be encoded by appropriately choosing a reduced MRF. This is called **Reduced Cutset Coding** (RCC); the rate for this coding method is stated precisely in Theorem 2.6.4 below.

**Theorem 2.6.4.** *Encoding Rate for RCC*

Let a MRF be defined on a graph  $G = (V, E)$ , and let  $p_G(\theta)$  be the distribution of the MRF. Let  $\mu$  be the moment coordinates corresponding to the parameter  $\theta$ ,  $L \subset V$  be a cutset, and  $\{\mathcal{C}_i\}$  be the set of components that result from removing the cutset. If the cutset is encoded using coding distribution  $p_{G_L}(\theta'_L)$ , then the overall rate  $R$  for RCC is,

$$R = \frac{|L|}{|V|} R_L + \frac{|V \setminus L|}{|V|} R_{V \setminus L},$$

where  $R_L$  and  $R_{V \setminus L}$  are the coding rates on the cutset and components respectively.

These are given by,

$$R_L = \frac{1}{|L|} \left[ H_G^L(\theta) + D(p_{G, \theta}^L \| p_{G_L, \mu_L}) + D(p_{G_L, \mu_L} \| p_{G_L, \theta'_L}) \right],$$

$$R_{V \setminus L} = \frac{1}{|V \setminus L|} H_G(X_{V \setminus L} | X_L) = \frac{1}{|V \setminus L|} \sum_{\mathcal{C}_i} H_G(X_{\mathcal{C}_i} | X_{\partial \mathcal{C}_i}).$$

*Proof.* The expression for  $R_L$  follows from Theorem 2.6.2 and Theorem 2.6.3. We omit the necessary steps and leave them to Reyes 2010 [23]. The last equality for  $R_{V \setminus L}$  follows from the MRF property of the components being conditionally independent from each other given the cutset <sup>18</sup>. □

---

<sup>18</sup>If two components were not conditionally independent, then they would not be disconnected. This can be easily shown for MRFs with only simple cliques, but is trickier with complex cliques. We discuss this further in Chapter 4 with the minimum radius condition.

It should be noted that by picking parameter  $\theta'_L$  so that it induces a moment-matching reduced MRF for  $\mu_L$ , the rate  $R_L$  is minimized. Finding these moment-matching parameters is not trivial. If the cutset forms an acyclic graph, then BP could be used. For cyclic cutsets, [23] proposes approximation via Loopy Belief Propagation [21], and Iterative Fitting or Scaling [5, 6].

# Chapter 3

## Learning and Precoding

In this chapter, we apply RCC to black and white images. We also seek to improve this method by introducing a precoding phase. This phase attempts to learn the best MRF parameter  $\theta \in \Theta$  for a given image. The encoding process would then involve encoding both the image itself and the “learned” parameters.

### 3.1 Learned RCC

The following steps give a high level description of the coding procedure,

1. Establish MRF and cutset specifications.
2. Learn parameter  $\theta$ .
3. Encode parameter  $\theta$ .
4. Encode cutsets.
5. Encode components.

Essentially, for each image, we tailor a coding distribution that is best suited for that image and encode the image accordingly. The decoding method would then be as follows,

1. Obtain MRF and cutset specifications.
2. Decode for  $\theta$ .
3. Decode cutset(s).
4. Decode remaining components.

We now give a precise breakdown of each step.

### 3.1.1 MRF and Cutset Specifications

Before learning or encoding, we specify the following,

1. MRF  $\mathcal{M} = (S, N, \mathcal{V})$
2. Cutset  $L \subset S$

First, a MRF is specified by the triple  $(S, N, \mathcal{V})$  of sites  $S$ , neighbourhood system  $N$ , and potentials  $\mathcal{V}$ <sup>1</sup>. The parameters  $\theta$  are intricately linked with the potentials  $\mathcal{V}$ ;  $\mathcal{V}$  can be seen as representing the parameters  $\theta$  and statistics  $t$  that give rise to a certain energy and distribution.

Second, the cutset  $L \subset S$  defines how the sites are to be segmented into components for RCC. Picking the cutset  $L$  also gives rise to an enumeration of graph

---

<sup>1</sup>We will address this issue in more detail in Chapter 4.

components  $\{\mathcal{C}_n\}$ . The enumeration and its ordering are not trivial and are important for proper decoding.

For example, in results we present in the next part of this Chapter, we may refer to the “Ising equipotential model” with “line spacing 3”. The former specifies MRF  $\mathcal{M}$ , and the latter specifies the cutset  $L$ . Together, they provide the identifying information on how the image was encoded.

### 3.1.2 Learning

It is assumed, for a given image (to be encoded), that there exists a MRF  $\mathcal{M}$  of which the image is a sample. Thus, the image can be used as a **training set** for learning this MRF. Specifically, for a set of sites  $S$  and a given neighbourhood system  $N$ , we are interested in knowing the parameter  $\theta$  for some statistics  $t$ . Once such an MRF is learned, it can be used to find coding distributions for the encoding process.

For the learning procedure, a variety of techniques could be used to estimate  $\theta$ . The results we present in this thesis will be concerned with the LS and MLE methods mentioned in Chapter 2.

### 3.1.3 Encoding Parameter, Cutset, and Components

The encoding process then involves 3 steps,

1. Encode  $\theta$  to a given precision.
2. Encode cutset  $L$  as a reduced MRF by arithmetic coding with the moment matching MRF.



3. Encode components  $\{\mathcal{C}_n\}$  conditioned on cutset  $L$  by arithmetic coding with the learned MRF.

The difference between this procedure and RCC is the encoding of parameter  $\theta$ . Encoding  $\theta$  is a fixed cost and becomes negligible for large images.

As mentioned in Chapter 2,  $L$  is encoded as a reduced MRF. In particular, note that to minimize inefficiency, it is desirable to identify the moment matching reduced MRF on  $L$  - this equates to another learning procedure of sorts to identify the parameters  $\theta_L$  that give rise to  $\mu_L$ . This would be a one time procedure for each specified MRF  $\mathcal{M}$  and can be pre-computed.<sup>2</sup> In this thesis, we do not make explicit attempts to find the moment matching reduced MRF on  $L$ . Instead, we simply encode  $L$  using some standard method (JBIG2) as a reduced MRF and incur any inefficiencies. For the components  $\{\mathcal{C}_n\}$ , arithmetic coding is made possible by clustering and BP.

### 3.1.4 Some Remarks About the Decoder

We assume the following decoder structure.

First, the encoder and decoder must have a pre-established agreement regarding the MRF  $\mathcal{M}$  and the cutset  $L$ . That is, the choice of MRF and cutset are not made dynamically during encoding, otherwise we would need to additionally encode the MRF specification, as well as the locations of the cutset pixels. This assumption is made apparent by the step (1) of encoding/decoding process.

Second, the encoder and decoder must have some pre-established agreement regarding the ordering MRF encoding of the components  $\{\mathcal{C}_n\}$ . Otherwise, we would

---

<sup>2</sup>The main concern here is computation. This step could be eliminated as an encoding time cost if say images to be encoded were learned and classified into a “bin” of MRFs. We discuss this later in Chapter 5.

need to encode the location of the components as well.

Finally, it is natural to say that any other procedural elements of the encoding process such as ordering of the parameter vector  $\theta$  is known to the decoder.

## 3.2 Estimation and Encoding

With gradient ascent, it is not clear if ML estimation yields the global maximizer. Thus, it is important to ask if the log likelihood is strictly concave. While not true for MRFs with large neighbourhoods or large number of cliques, strict concavity holds for MRFs with small neighbourhoods or few cliques.<sup>3</sup> For example, the results shown later this chapter are based on the Ising 4-point neighbourhood with a single parameter to estimate. Figures 3.10- 3.16 show that in this case, the log likelihood is strictly concave. Comparably, for MRFs with large neighbourhoods or large numbers of cliques, it becomes difficult to judge the concavity of the log likelihood due to high dimensionality of the parameter vector  $\theta$ . This is the case for results shown later in Chapter 5, where it is clear that the log likelihood is not strictly concave.

Also, it is not clear how many synthesized samples are needed to ensure that data collected from the samples reflect the statistical expected value. Without going into too much detail, in this thesis we choose the number of samples to generate based on the size of the MRF neighbourhood and neighbourhood system.<sup>4</sup>

We now present some experimental data for this coding procedure. The cutset we use are horizontal strips, 1-pixel in thickness, that segment the image into slices. We refer to the thickness of these slices as the **line spacing**. We then encode each

---

<sup>3</sup>By “small”, we mean no larger than 4-point Ising.

<sup>4</sup>Larger neighbourhood require more samples and larger neighbourhood systems require fewer samples.

slice by generating clusters 1-pixel in width, and running the BP algorithm on these clusters.

### 3.2.1 Using LS Estimation



Figure 3.1:  $512 \times 512$  bilevel test images.

In the following experiments, we use an Ising equipotential model with no external field. Recall from Chapter 2 that this is a 4-point neighbourhood system with  $\theta = \{\theta_1, \theta_2\}$ , where  $\theta_1 = 0$  (no external bias on nodes for a particular phase value), and where every edge potential function is governed by  $\theta_2$  (equipotential).

For test image “opp2” in Figure 3.1 (a), we get a LS estimate for  $\theta_2$  around 1.22. This yields a coding rate of approximately 0.093 bits per pixel with line spacing set at 2. If equipotential is not assumed and we use instead  $\theta = \{\theta_1, \theta_2, \theta_3\}$ , where  $\theta_2$  and  $\theta_3$  are the parameters for vertical and horizontal edge potentials respectively, then LS estimation gives  $\theta_1 = 1.19$ ,  $\theta_2 = 1.26$ , and a coding rate around 0.093 (again line spacing 2). This indicates that the image does not have edges correlated in any particular direction.

Now consider an image which does have “edges” biased in a particular direction.

Figure 3.1 (b) is a “barcode” where the lines are generated by Bernoulli(0.75) for white. With LS estimation, we get  $\theta_2 = 8.84$ ,  $\theta_3 = -0.0073$ , and a coding rate in the  $10^{-16}$ 's with line spacing 2. This seems almost unrealistically low at first glance, but we should note that this is only the rate on the “slices”. Cutset coding treats these slices as MRF's conditioned on cutsets. In this case, given the potentials on vertical edges, when we condition on the upper and lower boundaries (cutsets) of the slices, the observed values on these boundaries essentially dictate that the pixel values within the slices must be of a certain configuration with such high probability that it is virtually a dirac distribution. Thus, it is not so hard to believe that such a low code rate could be achieved on the “slices”. Heuristically, one might ask why we would want to use cutset coding at all in this example - a more efficient approach might be to just encode a single row of the image. Here, we note that doing so actually is no different than encoding the entire image using cutset coding without line spacing (i.e. no cutset). Thus, it also becomes clear that the choice of cutset play a significant role in the performance of coding.

### 3.2.2 Using ML Estimation

We also run this procedure with additional images shown in Figure 3.2 (images taken from USC-SIPI database [18]) and attain the following results - we include the JBIG2 compression results for comparison. Again, the Ising equipotential model with no external field is used. The cutset is composed again of horizontal strips with some line spacing.

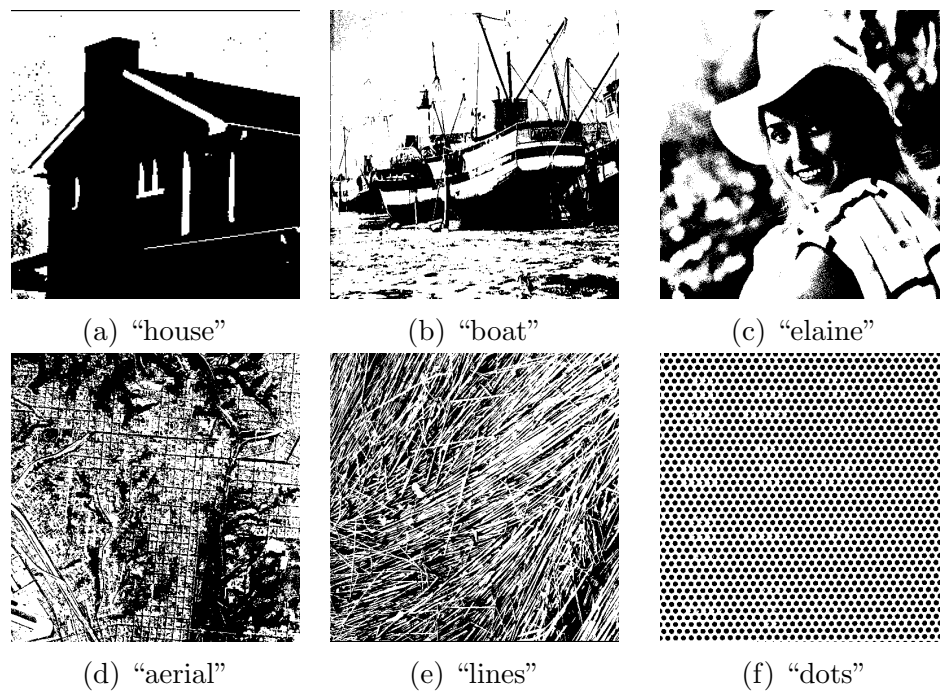
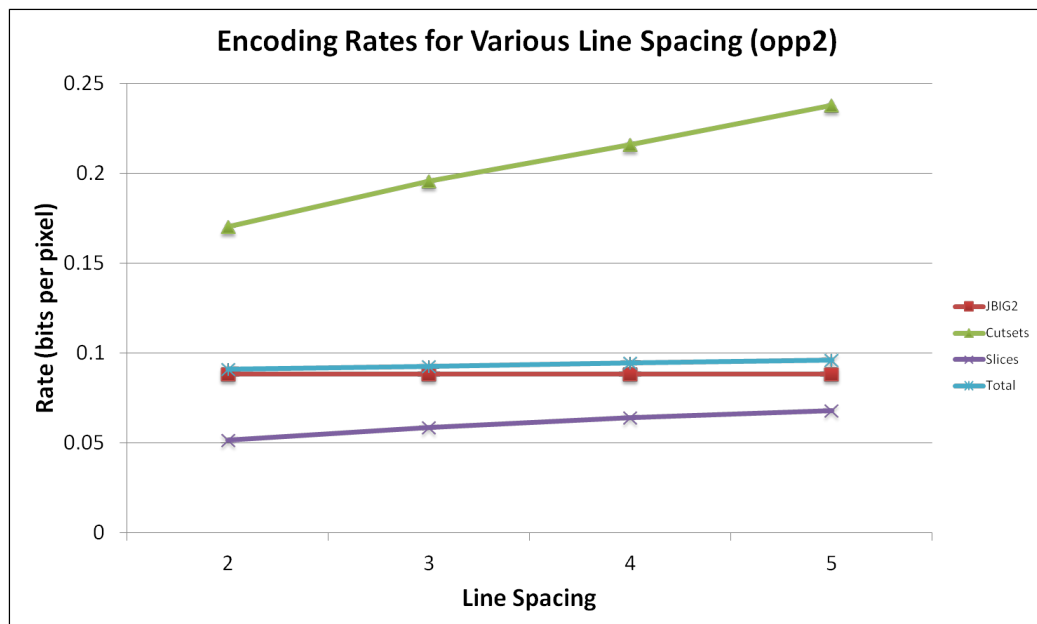
Figure 3.2:  $512 \times 512$  bilevel test images.

Table 3.1 below shows a comparison between the performance of encoding the images using RCC, and encoding them with JBIG2. All rates are shown in bits per pixel. Also, JBIG2 compression is used to encode the cutset (not to be confused with the comparison of overall encoding rate comparison with RCC and JBIG2). This means that we do not encode the cutset with a moment matching reduced MRF, and so we incur all the inefficiencies of doing so; as shown in the results below, the cost of encoding the cutset rises with increased line spacing.

Image	$\theta_2$	RCC	JBIG2
“opp2”	0.91	0.0911	0.0882
“house”	0.85	0.1067	0.1005
“boat”	0.62	0.2229	0.2288
“elaine”	0.59	0.2557	0.2340
“aerial”	0.44	0.5878	0.6317
“lines”	0.40	0.6886	0.6237
“dots”	0.52	0.2013	0.0648

Table 3.1: MLE parameters and code rates for test images from Figure 3.2

The following Figures show the coding rates on the cutsets, the component, and the average RCC rate.

Figure 3.3: Encoding rates for “opp2” with  $\theta_2 = 0.91$ .

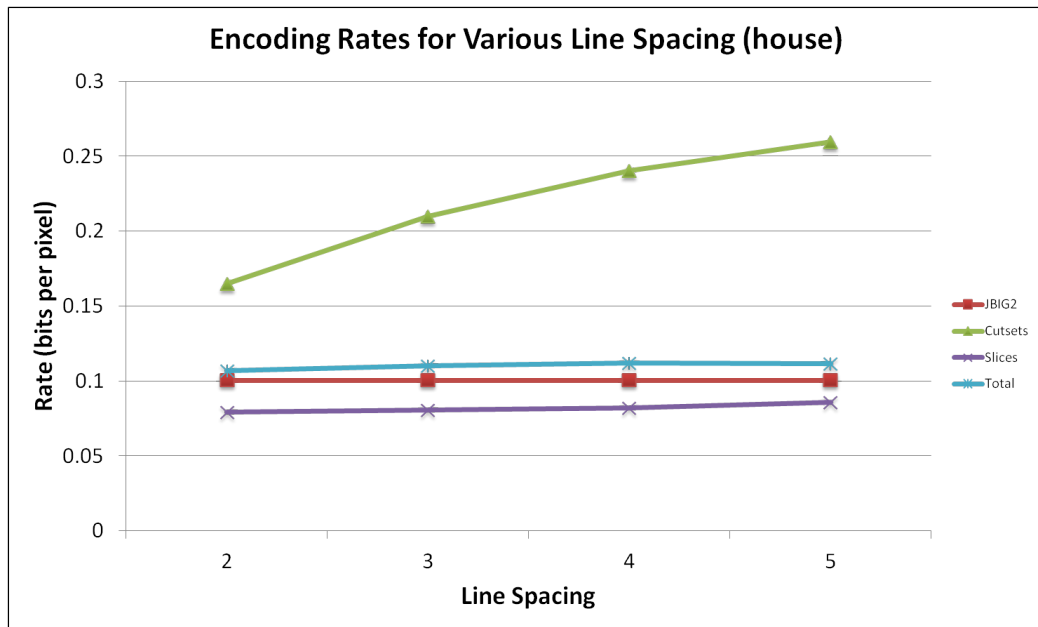


Figure 3.4: Encoding rates for “house” with  $\theta_2 = 0.85$ .

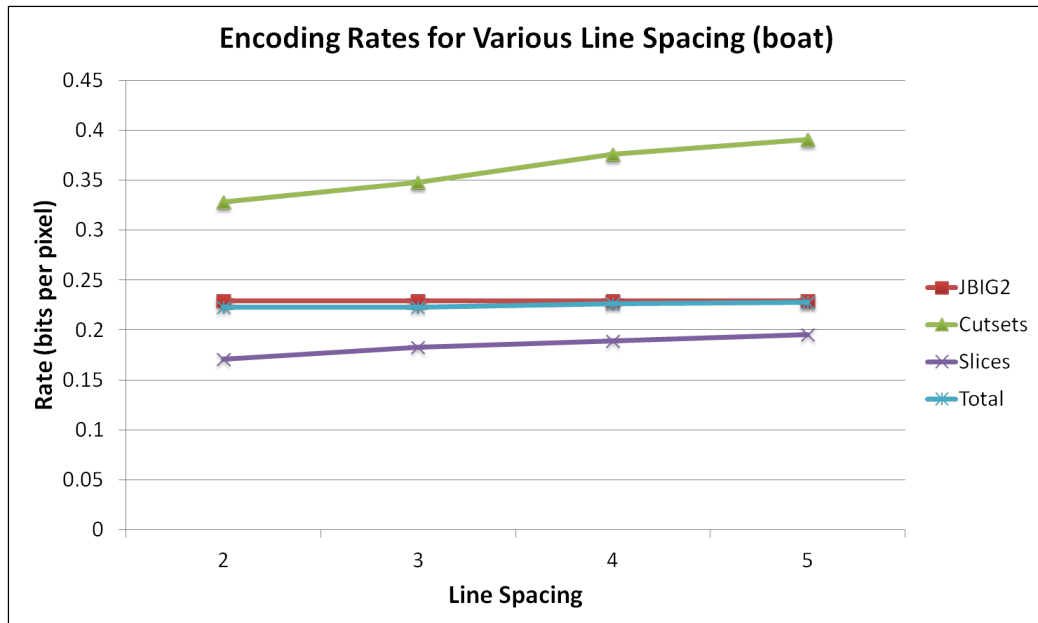
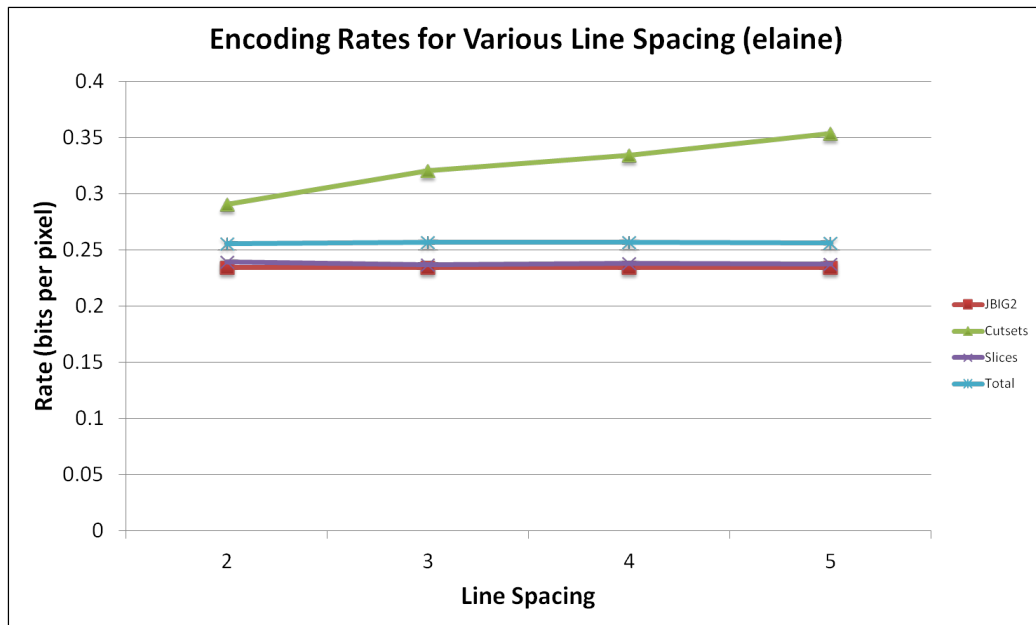
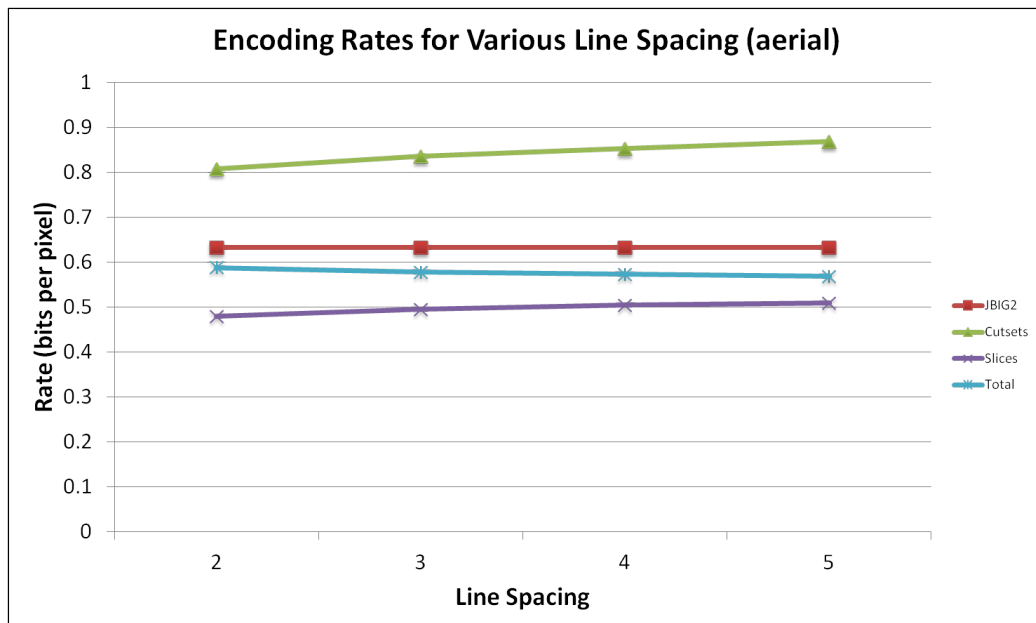
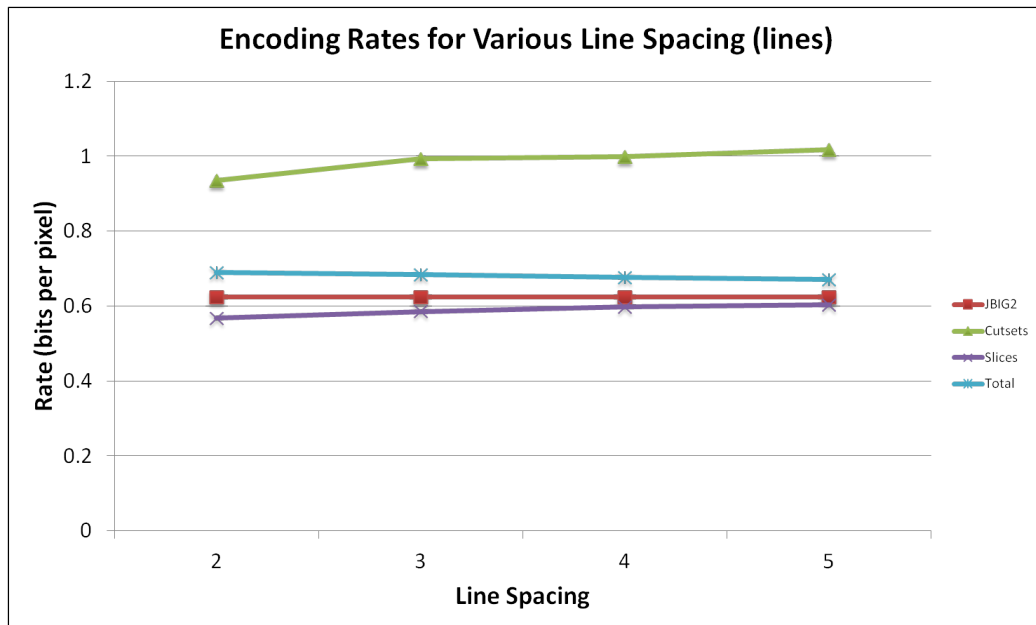
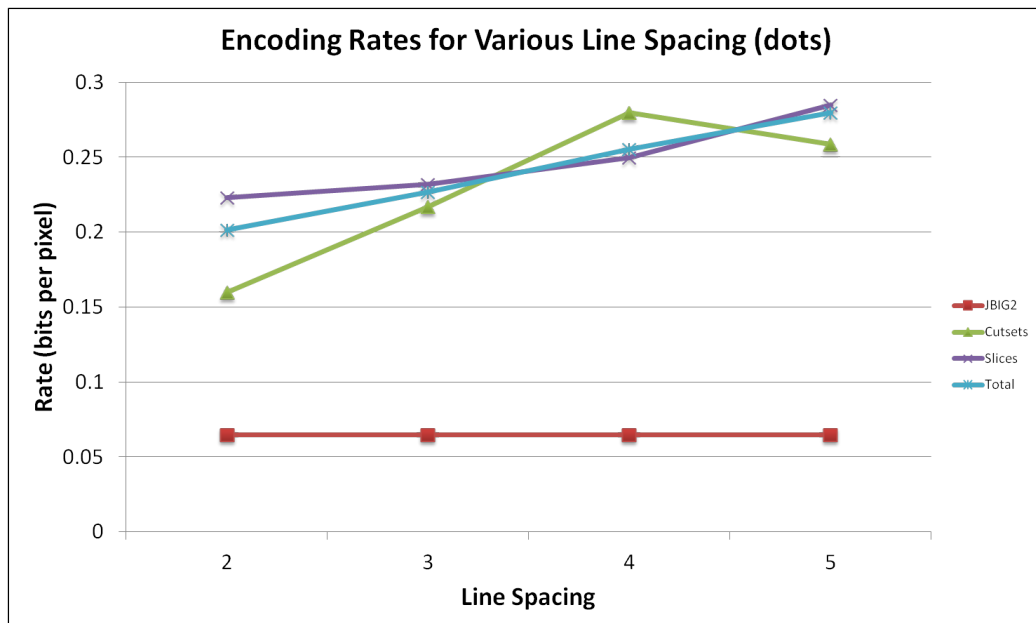


Figure 3.5: Encoding rates for “boat” with  $\theta_2 = 0.62$ .

Figure 3.6: Encoding rates for “elaine” with  $\theta_2 = 0.59$ .Figure 3.7: Encoding rates for “aerial” with  $\theta_2 = 0.44$ .



Figure 3.8: Encoding rates for “lines” with  $\theta_2 = 0.40$ .Figure 3.9: Encoding rates for “dots” with  $\theta_2 = 0.52$ .

As we can see, test images “house”, and “boat” perform somewhat well under this

coding scheme. This is most likely due to the images' similarities to the "blobby" nature of the Ising model (as seen in Chapter 2).

On the other hand, images "aerial" and "lines" perform poorly. This might largely be due to the fact that patterns exhibited in the images are of varying scale. Looking at the "aerial" photograph, we notice that there are blocks separated by a grid of streets. The streets show up largely as solid black lines a few pixels wide, while the blocks are largely white with a very textural pattern of black interspersed within. Individually, these blocks appear almost like a typical Ising image we saw in Chapter 2. Since the "blobs" within these blocks are quite small, this is suggestive of a high number of phase switches, hence a lower  $\theta_2$  (neighbouring nodes being less correlated). But we also notice that in the upper, lower right, and lower left regions of the image, we have large areas of black and white; this is suggestive of a higher  $\theta_2$  value. This means that estimating a "good"  $\theta$  essentially involved a trade-off between the sections of the image with higher, and lower energy - the greater the spread, the poorer we expect the performance to be. We note that in "boat" and "elaine", similar behaviour is also noted, but to a lesser degree <sup>5</sup>. We see this also in "lines".

Additionally, in "aerial", the "jagged" ways in which the black and white parts interact do not exhibit any particular directionality, and hence can appear more "chaotic" under this model. It is interesting to note that for LS estimation, this means that the relative abundance of neighbourhood configurations are not largely different, suggesting relatively equal energy across most configurations. In "lines", while the image displays a general directionality, this directionality is not captured by a 4-point neighbourhood. Thus, it might be better to consider to larger neighbourhoods

---

<sup>5</sup>More "blobs" and less "noise". We note in the case of "elaine" that the centre left portion of the image appear like random noise (uniform Bernoulli(0.5) pixels), high in entropy, and cannot be good for coding under a model where "blobs" are expected.

and cliques.

Lastly, we consider the test image “dots” where we have a very distinct pattern of “blobs”. Naturally, the code rate is better here. However, we should also note from Figure 3.16 (see below) that the ML estimator did not perform very well here. We should remember that the ML procedure relies on moment matching at each step of the gradient descent. In this case, the consistent pattern within the image actually matches well under a wider range of  $\theta_2$  - thus, it is hard to find the best estimate. This also suggests that the MRF here is poor in capturing the type of interactions in test images such as “dots”; similar, but poor matching over most  $\theta_2$ . Again, a relevant question we might ask is if larger neighbourhoods and cliques might be able to better “capture” this pattern. We discuss this in Chapters 4 and 5.

Making a comparison with M. Reyes’ results, he achieved a coding rate of roughly 0.065 bits per pixel (bpp) using  $\theta_2 = 0.6$  [23]. In our case, with a LS estimate of  $\theta_2 = 1.21$ , a rate of roughly 0.053 bpp was obtained. With a ML estimate of  $\theta_2 = 0.92$ , a rate of 0.051 bpp was obtained. Thus, even with a quick estimation procedure added to RCC, some significant improvement in rate can be made. Note that the rate comparisons here are being made strictly between the encoding of segmented components - we do this since our technique of encoding the cutset differs from that of Reyes’. It should also be noted that ML estimate is significantly more costly in computation time, though this does not necessarily become restricting since the estimation procedure could potentially be done “offline”.

When comparing the RCC results with that of JBIG2, with the exception of “dots” we note that the overall encoding rates were relatively on par with JBIG2 compression. There are two reasons that can explain this anomaly case. First, JBIG2 specializes in

compressing bi-level images mainly by extracting a combination of set-pattern data, and halftone regions [19].<sup>6</sup> As such, “dots” can be considered as an image that JBIG2 should perform extremely well on due to its symbol-like or strictly patterned nature. In many ways, this also explains why the compression rate is so poor on “arial” and “lines”; the images lack both pattern and halftone regions. Second, there is an observable periodicity within “dots”. This accounts for the the drop in cost of encoding cutset at line spacing 5 - see 3.9. Apart from contributing to the good performance of JBIG2, such image structures when modelled with a MRF with small neighbourhoods yield training data that suggest the abundance of virtually every kind of neighbourhood configuration, i.e. high entropy. Hence, it might be the case that the choice of MRF model here is particularly poor. Overall, since JBIG2 is a specialized bi-level encoding method, it is rather encouraging to note that RCC can be competitive and yet maintain its versatility in being extendable to greyscale or colour images as well.

Finally, before we end the discussion of above results, we note that to understand how close the proposed encoding method comes to theoretical bounds, ideally a comparison should be made between the encoding rates and the entropy rate of the source. Unfortunately in the case of large MRFs like here, the entropy rate is computationally intractable due to the partition function. Instead, the JBIG2 results can be considered as a good benchmark for encoding bi-level images.

---

<sup>6</sup>Can be intuitively understood to be structured visual artefacts in the former, and “gradient” areas in the latter (visual effect generated by varying relative density of black and white pixels).

### 3.2.3 Parameter Sensitivity

Regarding the encoding parameter  $\theta$ , we ask whether or not the coding rate is sensitive to deviations in an appropriate  $\theta$ , and if so, how precise need  $\theta$  be. Figure 3.10, Figures 3.11- 3.16 illustrate that the sensitivity is not high and we do not need to commit too much resource to encoding the parameter.

The data was gathered by taking the ML estimate, and encoding the image by increments above and below the estimate. One common characteristic we note for the figures below is the asymmetric nature of the curves. This is because  $\theta_2 = 0$  in this case indicates no “edge” interactions, and effectively means that the sites are independent of each other. Under such a scenario, since there are no relationships to exploit, for a binary phase space, we would expect something along the lines of 1 bit per pixel. Thus, while there might be a wide range of parameters that “well-capture” the features of a particular image, as  $\theta$  approaches 0, it is natural for the curves to approach the rate for coding independent sites.

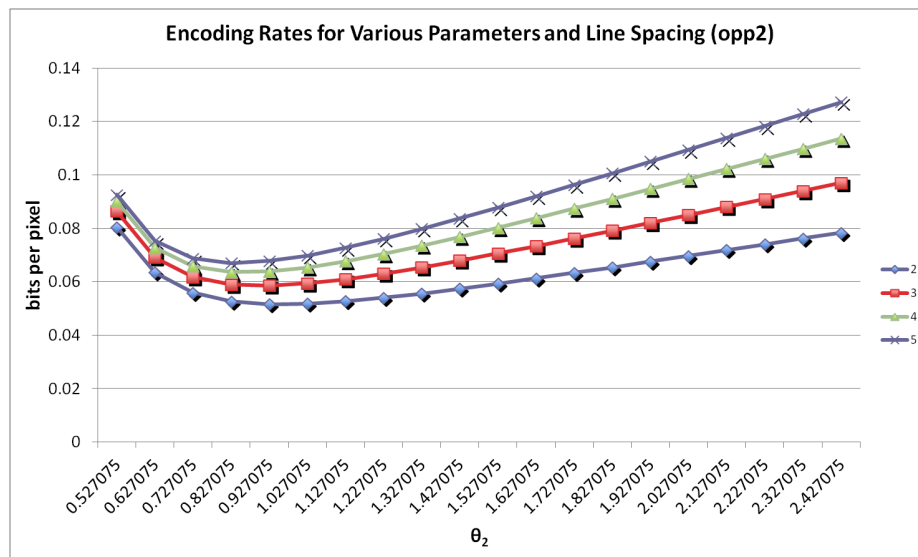


Figure 3.10: Encoding rates for “opp2” using various  $\theta_2$ , and line spacings.

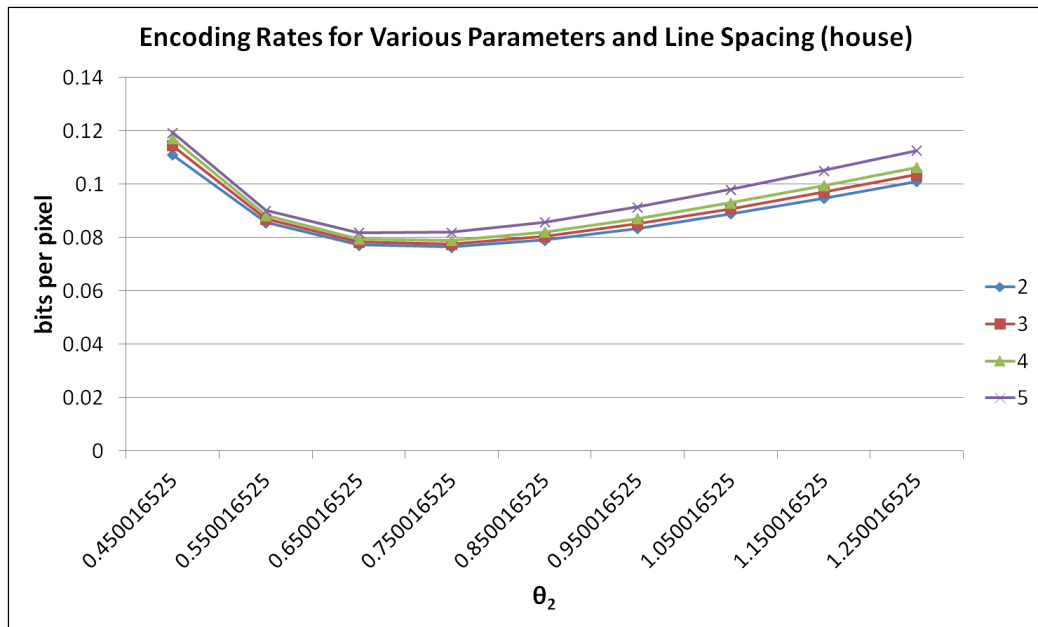


Figure 3.11: Encoding rates for “house” using various  $\theta_2$ , and line spacings.

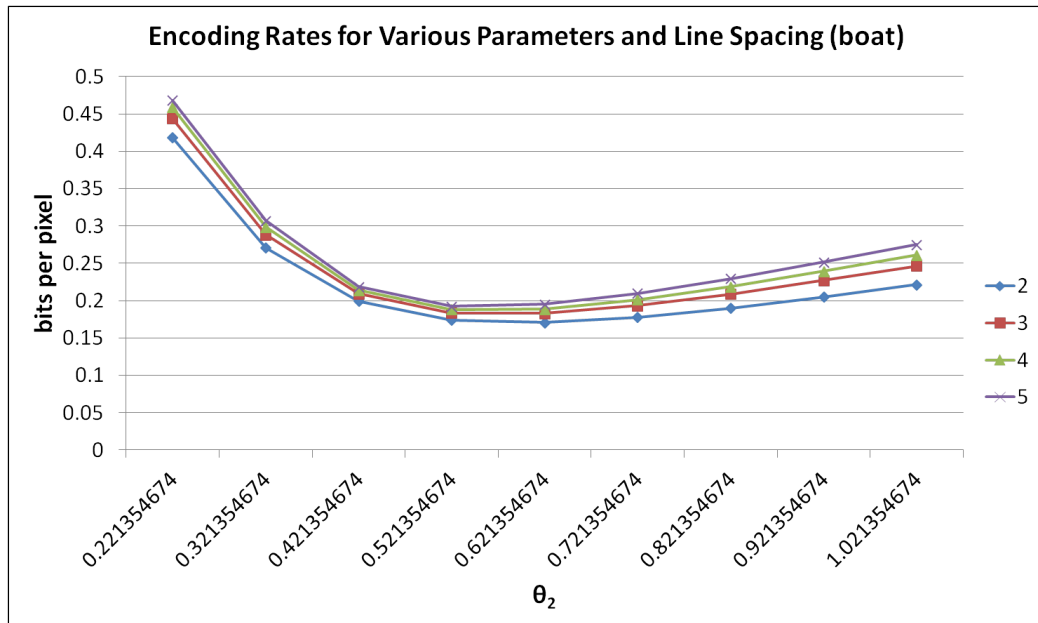


Figure 3.12: Encoding rates for “boat” using various  $\theta_2$ , and line spacings.

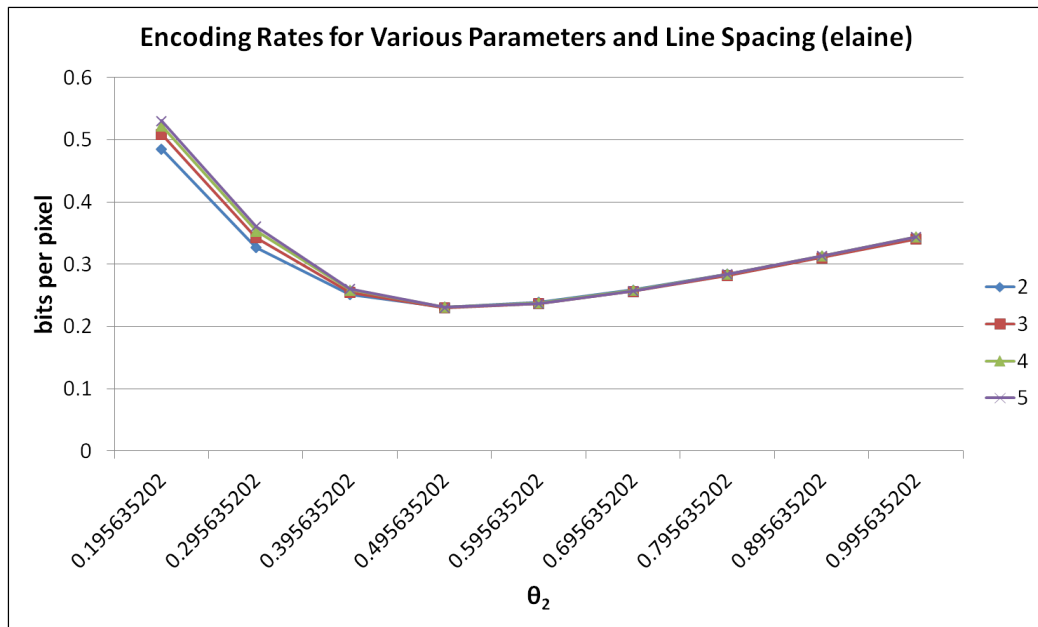


Figure 3.13: Encoding rates for “elaine” using various  $\theta_2$ , and line spacings.

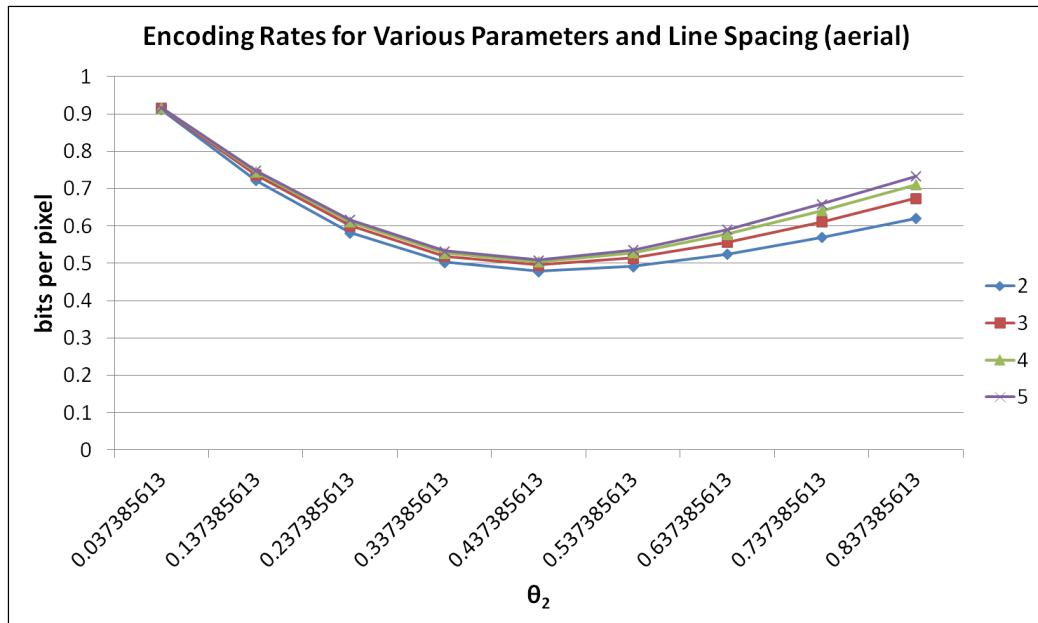


Figure 3.14: Encoding rates for “aerial” using various  $\theta_2$ , and line spacings.

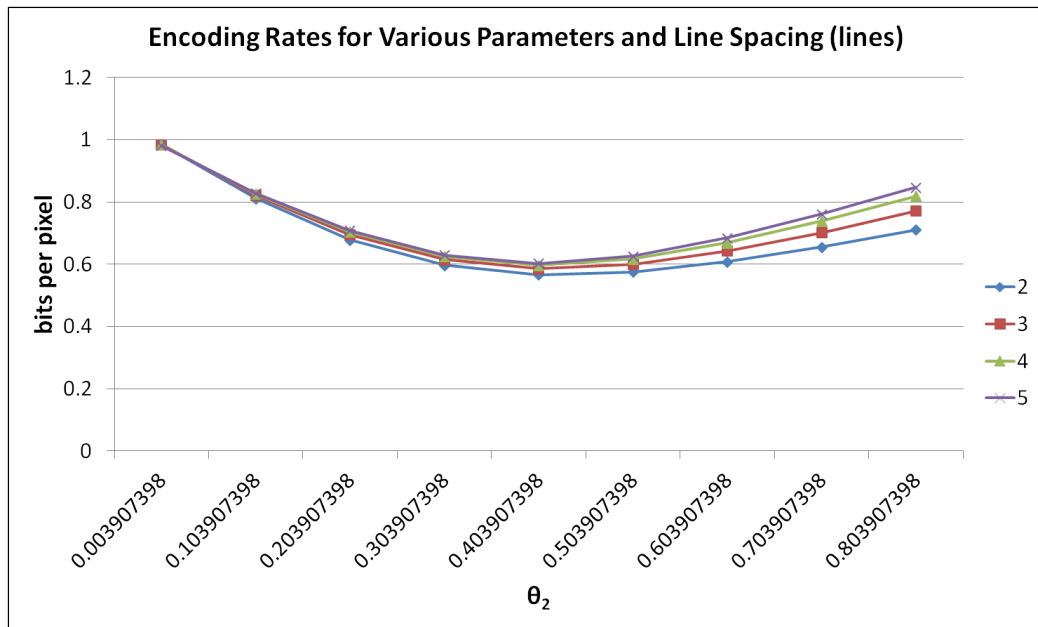


Figure 3.15: Encoding rates for “lines” using various  $\theta_2$ , and line spacings.

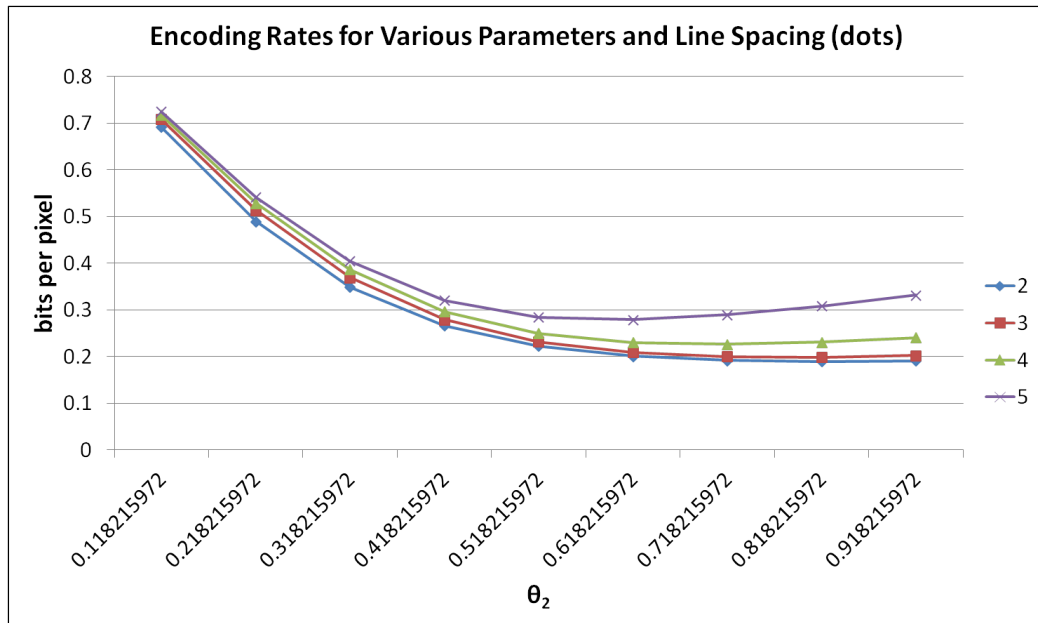


Figure 3.16: Encoding rates for “dots” using various  $\theta_2$ , and line spacings.



# Chapter 4

## MRFs with Complex Cliques

Chapter 3 showed that good compression can be achieved by learning the appropriate MRF parameters. Since this was done mainly for the simple cliques (pairwise interactions) of the “Ising model”, it’s natural to ask how well this method performs on MRF’s with complex cliques. To our knowledge, this topic has not been explored in the past.

An immediate relevant question regarding complex cliques is whether or not the prescribed estimation and inference methods mentioned in Chapter 2 work.

For example, where complex cliques are concerned, BP breaks down simply by the fact that it is only capable of accounting for interaction between at most 2 nodes. With clustering, we saw that BP in the traditional sense can be made to work by grouping sites together and creating an acyclic graph. As it turns out, though less intuitive, similar modifications can be made in the case of complex cliques. We will devote the rest of this chapter to explain the necessary changes in detail. In Chapter 5, we will discuss some experimental results.

## 4.1 Some Preliminaries

### 4.1.1 Some Definitions

To start, recall that MRF's and Gibbs fields are equivalent, and that Gibbs fields can be specified by the Gibbs potential which are given by some neighbourhood system and the associated cliques. Knowing this, we can describe an MRF,  $\mathcal{M}$ , by the 3-tuple,

$$\mathcal{M} = (S, N, \mathcal{V})$$

where  $S$  is the set of nodes (sites),  $N$  is the neighbourhood family, and  $\mathcal{V}$  is the Gibbs potential relative to  $N$ . Given MRF  $\mathcal{M} = (S, N, \mathcal{V})$ , we also define the following,

- A **cluster** or **supernode** is a subset of nodes  $A \subset S$ .
- The set of cliques in a cluster  $A$  is denoted by,

$$\mathcal{C}_A = \{C : C \subset A, C \text{ is a clique}\}.$$
<sup>1</sup>

- The set of all cliques “between” clusters  $A$  and  $B$  is denoted by,

$$\Delta_{A,B} = \{C : C \cap A \neq \emptyset, C \cap B \neq \emptyset, C \subset A \cup B, C \text{ is a clique}\}$$

Additionally, if  $A \cap B = \emptyset$ , and  $\Delta_{A,B}$  is non-empty, then  $\Delta_{A,B}$  is referred to as the **superedge** between  $A$  and  $B$ . We will also use the notation  $\Delta_{A,B,D}$  to denote the set of cliques “between” the three clusters  $A$ ,  $B$ , and  $D$ ; and so on

---

<sup>1</sup>Remember that cliques are *induced* by the neighbourhood family  $N$  of a MRF

for any finite number of clusters.<sup>2</sup>

We are now ready to talk about *cluster graphs*.

### 4.1.2 Cluster Graphs

A **cluster graph induced by MRF**  $\mathcal{M}$  is a pair  $\mathcal{G} = (\mathcal{A}, \Delta_{\mathcal{A}})$ , where  $\mathcal{A} = \{A_1, \dots, A_n\}$  is a set of disjoint supernodes that form a partition of  $S$ ,

$$A_i \cap A_j = \emptyset, i \neq j$$

$$S = \bigcup_{i=1}^n A_i$$

and  $\Delta_{\mathcal{A}}$  is the set of all superedges exist between the supernodes in  $\mathcal{A}$ . The reason we refer to the cluster graph as being induced by MRF  $\mathcal{M}$  is because the cluster graph very much depends on the structure of the MRF. From this point forth, all the cluster graphs we deal with will be induced by some MRF defined a priori, and thus we'll simply leave out the “induced” part. The following are some definitions, and terminology about cluster graphs,

- The set of **cliques in cluster graph**  $\mathcal{G}$  is defined as,

$$\mathcal{C}_{\mathcal{G}} = \left( \bigcup_{A_i \in \mathcal{A}} \mathcal{C}_{A_i} \right) \cup \left( \bigcup_{\Delta_{A_i, A_j} \in \Delta_{\mathcal{A}}} \Delta_{A_i, A_j} \right).^3$$

- A cluster graph  $\mathcal{G} = (\mathcal{A}, \Delta_{\mathcal{A}})$  is said to satisfy the **minimum-radius** condition

---

<sup>2</sup>Note that the condition  $C \subset A \cup B$  is important in ensuring that the clique  $C$  is contained strictly by  $A$  and  $B$ . This prevents repeat repeat entries in  $\Delta_{A, B}$  and  $\Delta_{A, B, D}$ , i.e.  $\Delta_{A, B} \cap \Delta_{A, B, D} = \emptyset$ .

<sup>3</sup>Note that  $\mathcal{C}_{\mathcal{G}} \subseteq \mathcal{C}_S$

if,

$$\mathcal{C}_S = \mathcal{C}_G$$

- Supernodes  $A_i, A_j \in \mathcal{A}$  are **cluster neighbours** if  $\Delta_{A_i, A_j} \in \Delta_{\mathcal{A}}$ . We will use  $\gamma A$  to denote the set of all neighbours of  $A$ , i.e.  $\gamma A = \{B \in \mathcal{A} : \Delta_{A, B} \in \Delta_{\mathcal{A}}\}$ .
- A **cluster path** is a sequence of supernodes  $\{B_i \in \mathcal{A}\}$  where successive supernode pairs  $(B_m, B_{m+1})$  are neighbours.
- A **connected cluster graph** is a graph where every pair of supernodes can be linked by some path.
- A **cluster cycle** is a cluster path that starts and ends on the same super node. If no cycles exist within a cluster graph, then it is said to be a **acyclic cluster graph**. A **cluster tree** is simply a connected acyclic cluster graph.
- For a connected cluster graph, if removing some superedge  $\Delta_{A, B}$  between supernodes  $A$  and  $B$  creates a disconnected cluster graph, then  $\Delta_{A, B}$  is called a **super-cut-edge** and we use the notation  $\mathcal{G}_{A \setminus B}$  and  $\mathcal{G}_{B \setminus A}$  to denote the components containing cluster  $A$  and cluster  $B$  respectively (this is completely analogous to the graph notation discussed in Chapter 2). We will also denote the set of sites on these subgraphs  $S_A$  and  $S_B$  respectively.

## 4.2 Useful Lemmas and Corollaries

**Lemma 4.2.1.** *If cluster graph  $\mathcal{G} = (\mathcal{A}, \Delta_{\mathcal{A}})$  is acyclic, then it satisfies the minimum-radius condition. i.e.  $\mathcal{C}_S = \mathcal{C}_G$ .*

*Proof.* We proceed by proving the contrapositive. Assume that the minimum-radius condition is not satisfied, then  $\exists C$  such that  $C \in \mathcal{C}_S$ , but  $C \notin \mathcal{C}_G$ . By definition,  $\mathcal{C}_G$  accounts for all cliques in two categories: (1) cliques within a cluster  $A_i \in \mathcal{A}$ , and (2) cliques between any two clusters  $A_i, A_j \in \mathcal{A}$ . Thus,  $C$  must contain sites from at least 3 different clusters - denote these  $A_i, A_j$ , and  $A_k$ . Arbitrarily select sites  $a_i, a_j, a_k \in C$ , where  $a_i \in A_i, a_j \in A_j, a_k \in A_k$ . By definition of cliques,  $a_i, a_j$ , and  $a_k$  are pairwise neighbours of each other, and so  $\{a_i, a_j\}$ ,  $\{a_j, a_k\}$ , and  $\{a_k, a_i\}$  are valid cliques. This means  $\Delta_{A_i, A_j}$ ,  $\Delta_{A_j, A_k}$ , and  $\Delta_{A_k, A_i}$  are non-empty, and are thus valid superedges that form a cycle -  $\mathcal{G}$  cannot be acyclic.  $\square$

It is worthwhile to note that the converse is not always true. This can be illustrated by the example of a *doughnut* graph shown in Figure 4.1. For the Ising model with a 4-point neighbourhood, it is easy to see that while  $\mathcal{C}_S = \mathcal{C}_G$ , the graph remains cyclic.

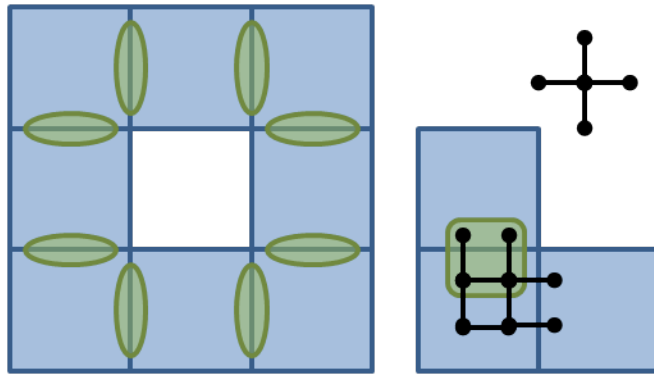


Figure 4.1: A cluster graph of 8 supernodes arranged in a *doughnut* manner. The 4-point Ising neighbourhood system is used.

Before continuing, it's worth it to provide a brief explanation of why the minimum radius condition is interesting. It is interesting mainly in that it is a necessary condition for a cluster graph to be acyclic. As we shall soon see, one of the main problems of cluster graphs is keeping track of cliques and making sure that certain

clique configurations are not omitted in computations. When omitted, the results is either wrong marginalization, dysfunctional BP, or unending BP from cyclic cluster graphs. By understanding the minimum radius condition, these problems can be eliminated, and the task of ensuring that a cluster graph is acyclic can be reduced to simply ensuring that the minimum radius condition is met, and ensuring there are no cycles in the traditional graph theory sense.

We now present an interesting result that tells us that if cluster graph  $\mathcal{G}$  satisfies the minimum-radius condition, then its subgraphs satisfy the condition as well.

**Lemma 4.2.2.** *Let  $\mathcal{G} = (\mathcal{A}, \Delta_{\mathcal{A}})$  be a cluster graph satisfying the minimum-radius condition, and let  $\Delta_{A,B} \in \Delta_{\mathcal{A}}$  be a super-cut-edge, then the subgraphs  $\mathcal{G}_{\mathcal{A}\setminus B} = (\mathcal{A}_A, \Delta_{\mathcal{A}_A})$  and  $\mathcal{G}_{B\setminus A} = (\mathcal{A}_B, \Delta_{\mathcal{A}_B})$  also satisfy the minimum-radius condition (on  $S_A$  and  $S_B$  respectively). That is,*

$$\mathcal{C}_{S_A} = \mathcal{C}_{\mathcal{G}_{\mathcal{A}\setminus B}}$$

$$\mathcal{C}_{S_B} = \mathcal{C}_{\mathcal{G}_{B\setminus A}}$$

*Proof.* Again, we proceed by proving the contrapositive. Assume that  $\mathcal{C}_{S_A} \neq \mathcal{C}_{\mathcal{G}_{\mathcal{A}\setminus B}}$ , that is  $\exists C$  such that  $C \in \mathcal{C}_{S_A}$ , but  $C \notin \mathcal{C}_{\mathcal{G}_{\mathcal{A}\setminus B}}$ . Since,

$$\mathcal{A}_A \subset \mathcal{A}$$

$$\Delta_{\mathcal{A}_A} \subset \Delta_{\mathcal{A}}$$

we have that  $C \notin \mathcal{C}_{\mathcal{G}}$ , meaning  $\mathcal{C}_S \neq \mathcal{C}_{\mathcal{G}}$ , and so  $\mathcal{G}$  cannot satisfy the minimum-radius condition. □

The following Lemma shows that mutual exclusive clusters induce mutually exclusive clique sets.

**Lemma 4.2.3.** *For clusters  $A, B \subset S$ , the following statements are equivalent*

$$(a) \ A \cap B = \emptyset$$

$$(b) \ \mathcal{C}_A \cap \mathcal{C}_B = \emptyset$$

$$(c) \ \mathcal{C}_A \cap \Delta_{A,B} = \emptyset$$

$$(d) \ \mathcal{C}_B \cap \Delta_{A,B} = \emptyset$$

*Proof.* (a)  $\iff$  (b) by definition of  $\mathcal{C}_A$ . We show (a)  $\iff$  (c) by showing the contrapositives. First, assume that  $\mathcal{C}_A \cap \Delta_{A,B} \neq \emptyset$ . This means  $\exists$  some clique  $C \subset A$  such that  $C \cap B \neq \emptyset$ , and thus  $A \cap B \neq \emptyset$ . Now assume that  $A \cap B \neq \emptyset$ . Since any singleton is a clique, by taking any node  $n \in (A \cap B)$ , we get that  $\{n\} \in \mathcal{C}_A$ , and  $\{n\} \in \Delta_{A,B}$ , and thus  $\mathcal{C}_A \cap \Delta_{A,B} \neq \emptyset$ . The proof for (a)  $\iff$  (d) follows the same steps, but uses  $\mathcal{C}_B$  instead.  $\square$

**Lemma 4.2.4.** *Total Cliques*

*Given clusters  $A, B \subset S$ , let  $A \cup B = D$ . Then we have that,*

$$\mathcal{C}_D = \mathcal{C}_A \cup \mathcal{C}_B \cup \Delta_{A,B} = \mathcal{C}_{A,B}$$

*Note that the last part of the equality is just a change in notation. As opposed to writing  $\mathcal{C}_{(A \cup B)}$ , we simply write  $\mathcal{C}_{A,B}$ .*

*Proof.* Take any clique  $C \subset D$ , then either  $C \subset A$ , that is  $C \in \mathcal{C}_A$ ; or  $C \subset B$ , that is  $C \in \mathcal{C}_B$ ; or  $C$  is neither in  $A$  nor  $B$ , but is in “between”  $A$  and  $B$ , that is  $C \cap A \neq \emptyset$  and  $C \cap B \neq \emptyset$ , and so  $C \in \Delta_{A,B}$ .  $\square$

By the above lemmas, we attain the following corollary useful for manipulating potentials.

**Corollary 4.2.1.** *Let  $A, B \subset S$  be disjoint, and let  $D = A \cup B$ , then,*

$$\prod_{C \in \mathcal{C}_D} \psi_C(x_D) = \prod_{C \in \mathcal{C}_A} \psi_C(x_A) \prod_{C \in \mathcal{C}_B} \psi_C(x_B) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_B)$$

where  $\psi_C(x_D) = \exp(V_C(x_D))$ .

*Proof.* By Lemma 4.2.4, we have that,

$$\prod_{C \in \mathcal{C}_D} \psi_C(x_D) = \prod_{C \in (\mathcal{C}_A \cup \mathcal{C}_B \cup \Delta_{A,B})} \psi_C(x_D)$$

Now since  $A \cap B = \emptyset$ , by Lemma 4.2.3, we have that,

$$\prod_{C \in \mathcal{C}_D} \psi_C(x_D) = \prod_{C \in \mathcal{C}_A} \psi_C(x_A) \prod_{C \in \mathcal{C}_B} \psi_C(x_B) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_B)$$

□

### 4.3 Clustering and Belief Propagation for MRF's with Complex Cliques

We now begin the derivations that will lead us to a modified BP algorithm on cluster graphs induced by an MRF with complex cliques. We first introduce the following.

**Definition 4.3.1.** *Supernode Self-potential*



Let  $\mathcal{G}$  a cluster graph. For a supernode  $A \in \mathcal{A}$ , we define the **supernode self-potential** of  $A$  to be,

$$\Psi_A(x_A) = \prod_{C \in \mathcal{C}_A} \psi_C(x_A)$$

**Definition 4.3.2.** *Superedge potential*

Let  $\mathcal{G}$  a cluster graph. For neighbouring supernodes  $A, B \in \mathcal{A}$ , we define the **superedge potential** of  $\Delta_{A,B} \in \Delta_A$  to be,

$$\Psi_{A,B}(x_A x_B) = \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_B)$$

This is analogous to the self and edge potentials defined previously in our discussion of BP for graphs of nodes and edges. Also, as the case with superedge notation, we'll use the modified notation of  $\Psi_{A,B,D}(x_A x_B x_D)$  to denote the product of potentials on cliques between more than two nodes. We now proceed to present the BP algorithm for cluster graphs.

### 4.3.1 Beliefs, Messages, and Recursion

Recall that the belief vector for a set of sites  $A \subset S$  is defined as  $Z_A = \{Z_A(x_A) : x_A \in \Lambda^A\}$  where,

$$Z_A(x_A) = \sum_{x_{S \setminus A}} \prod_{C \in \mathcal{C}_S} \exp(-V_C(x_A x_{S \setminus A}))$$

The definition of belief remains the same here (in the context of complex cliques). Recall also that  $Z_A(x_A) \propto p_A(x_A)$  since the sum essentially amounts to an unnormalized marginalization. Additionally, we define the following,

**Definition 4.3.3.** *Belief on Cluster Subgraphs*

Let  $\mathcal{G}$  be a cluster graph, and let  $\mathcal{G}_{A \setminus B}$  be the cluster subgraph containing cluster  $A$  after removing the super-cut-edge  $\Delta_{A,B} \in \Delta_A$ . Then the belief of cluster  $A$  on the subgraph  $\mathcal{G}_{A \setminus B}$  is defined to be  $Z_{A \setminus B} = \{Z_{A \setminus B}(x_A) : x_A \in \Lambda^A\}$  where,

$$Z_{A \setminus B}(x_A) = \sum_{x_{S_A \setminus A}} \prod_{C \in \mathcal{C}_{S_A}} \exp(-V_C(x_A x_{S_A \setminus A}))$$

We now define messages for BP on cluster graphs.

**Definition 4.3.4.** *Messages for Cluster Graphs*

Let  $\mathcal{G}$  be some cluster graph. For some super-cut-edge  $\Delta_{A,B} \in \Delta_A$ , the **message** from supernode  $B$  to supernode  $A$  is defined to be  $m_{B \rightarrow A} = \{m_{B \rightarrow A}(x_A) : x_A \in \Lambda^A\}$  where,

$$m_{B \rightarrow A}(x_A) = \sum_{x_B} \Psi_{A,B}(x_A x_B) Z_{B \setminus A}(x_B)$$

We now present a lemma which will help us find the necessary belief decomposition for the Belief Propagation Algorithm.

**Lemma 4.3.1.** *Cluster Subgraph Message Passing*

Let  $\mathcal{G}$  be a cluster graph, and  $\Delta_{A,B} \in \Delta_A$  be a super-cut-edge, then,

$$Z_A(x_A) = Z_{A \setminus B}(x_A) m_{B \rightarrow A}(x_A)$$

That is to say the belief of cluster  $A$  in  $\mathcal{G}$  is simply the belief of  $A$  on the subgraph  $\mathcal{G}_{A \setminus B}$  times the “message” passed from cluster  $B$  to  $A$  “across” the super-cut-edge (essentially accounting for the “potential” from the other cluster subgraph).<sup>4</sup>

---

<sup>4</sup>For some configuration  $x_A$  on  $A$ .

*Proof.* Since  $\Delta_{A,B} \in \Delta_A$  is a super-cut-edge on  $\mathcal{G}$ , the set of sites on the resulting subgraphs form a partition of all sites. That is,

$$S_A \cup S_B = S$$

$$S_A \cap S_B = \emptyset$$

and so by Corollary 4.2.1, we have,

$$\begin{aligned} Z_A(x_A) &= \sum_{x_{S \setminus A}} \prod_{C \in \mathcal{C}_S} \exp(-V_C(x_A x_{S \setminus A})) \\ &= \sum_{x_{S \setminus A}} \prod_{C \in \mathcal{C}_S} \psi_C(x_A x_{S \setminus A}) \\ &= \sum_{x_{S \setminus A}} \prod_{C \in \mathcal{C}_{S_A}} \psi_C(x_A x_{S \setminus A}) \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_A x_{S \setminus A}) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_{S \setminus A}) \\ &= \sum_{x_{S \setminus A}} \prod_{C \in \mathcal{C}_{S_A}} \psi_C(x_A x_{S \setminus A}) \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_{S \setminus A}) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_{S \setminus A}) \end{aligned}$$

Now notice that  $S \setminus A = S_B \cup (S_A \setminus A)$ , and so we can break down the sum as,

$$\begin{aligned} Z_A(x_A) &= \sum_{x_{S_B}} \sum_{x_{S_A \setminus A}} \prod_{C \in \mathcal{C}_{S_A}} \psi_C(x_A x_{S_B} x_{S_A \setminus A}) \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_{S_B} x_{S_A \setminus A}) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_{S_B} x_{S_A \setminus A}) \\ &= \sum_{x_{S_B}} \sum_{x_{S_A \setminus A}} \prod_{C \in \mathcal{C}_{S_A}} \psi_C(x_A x_{S_A \setminus A}) \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_{S_B}) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_{S_B}) \\ &= \left( \sum_{x_{S_A \setminus A}} \prod_{C \in \mathcal{C}_{S_A}} \psi_C(x_A x_{S_A \setminus A}) \right) \left( \sum_{x_{S_B}} \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_{S_B}) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_{S_B}) \right) \\ &= Z_{A \setminus B}(x_A) \left( \sum_{x_{S_B}} \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_{S_B}) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_{S_B}) \right), \end{aligned}$$

where we were able to separate the sums due to the fact that potential functions only depend the values of sites in the associated clique (as discussed in Chapter 2).

Now notice that  $S_B = B \cup (S_B \setminus B)$ , and so we can again break the sum into a double sum.

$$\begin{aligned}
Z_A(x_A) &= Z_{A \setminus B}(x_A) \left( \sum_{x_B} \sum_{x_{S_B \setminus B}} \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_B x_{S_B \setminus B}) \prod_{C \in \Delta_{A,B}} \psi_C(x_A x_B) \right) \\
&= Z_{A \setminus B}(x_A) \left( \sum_{x_B} \Psi_{A,B}(x_A x_B) \left( \sum_{x_{S_B \setminus B}} \prod_{C \in \mathcal{C}_{S_B}} \psi_C(x_B x_{S_B \setminus B}) \right) \right) \\
&= Z_{A \setminus B}(x_A) \left( \sum_{x_B} \Psi_{A,B}(x_A x_B) Z_{B \setminus A}(x_B) \right) \\
&= Z_{A \setminus B}(x_A) m_{B \rightarrow A}(x_A),
\end{aligned}$$

where again, the sums were separated based on the dependency of potential functions. □

Lemma 4.3.1 gives the basis for calculating the belief of a cluster by collecting messages from neighbouring nodes. In an acyclic cluster graph, every superedge is a super-cut-edge, and so we have the following theorem.

**Theorem 4.3.1.** *Belief Decomposition for Acyclic Cluster Graphs*

*Let  $\mathcal{G}$  be a connected acyclic cluster graph, and  $A$  be some cluster in the graph. Then,*

$$Z_A(x_A) = \Psi_A(x_A) \prod_{B \in \gamma A} m_{B \rightarrow A}(x_A)$$

*Proof.* First, notice that when  $|\gamma A| = 1$ , the result is equivalent to Lemma 4.3.1.

Since  $\mathcal{G}$  is connected, the cluster subgraph  $\mathcal{G}_{A \setminus B}$  generated by removing super-cut-edge  $\Delta_{A,B}$  contains only a single cluster - namely  $A$  - and so,

$$\begin{aligned}
 Z_A(x_A) &= Z_{A \setminus B}(x_A) m_{B \rightarrow A}(x_A) \\
 &= \left( \sum_{x_{S_A \setminus A}} \prod_{C \in \mathcal{C}_{S_A}} \psi_C(x_A x_{S_A \setminus A}) \right) m_{B \rightarrow A}(x_A) \\
 &= \left( \prod_{C \in \mathcal{C}_A} \psi_C(x_A) \right) m_{B \rightarrow A}(x_A) \\
 &= \Psi_A(x_A) m_{B \rightarrow A}(x_A)
 \end{aligned}$$

Now for  $|\gamma A| > 1$ , since  $\mathcal{G}$  is acyclic, notice that  $Z_{A \setminus B}(x_A)$  can be found by reapplying the results of Lemma 4.3.1 - this time to the subgraph  $\mathcal{G}_{A \setminus B}$ . That is, let  $\mathcal{G}_{A \setminus B} = (\mathcal{A}_A, \Delta_{\mathcal{A}_A})$ , and let  $D$  be a neighbouring cluster of  $A$  in the subgraph ( $D \in \mathcal{A}_A, \Delta_{A,D} \in \Delta_{\mathcal{A}_A}$ ), then we have that,

$$Z_{A \setminus B}(x_A) = Z'_{A \setminus D}(x_A) m_{D \rightarrow A}(x_A)$$

where  $Z'_{A \setminus D}(x_A)$  is belief calculated on a subgraph of  $\mathcal{G}_{A \setminus B}$ , *not* of  $\mathcal{G}$ . Thus it is possible to recursively apply Lemma 4.3.1 until only one neighbour remains, and so,

$$Z_A(x_A) = \Psi_A(x_A) \prod_{B \in \gamma A} m_{B \rightarrow A}(x_A)$$

□

The recursive formula then follows.

**Theorem 4.3.2.** *Message Recursion on Cluster Graphs*

Let  $\mathcal{G}$  be a connected acyclic cluster graph satisfying the minimum-radius condition. Messages can be expressed recursively by,

$$m_{B \rightarrow A}(x_A) = \sum_{x_B} \Psi_{A,B}(x_A x_B) \Psi_B(x_B) \prod_{D \in (\gamma_B) \setminus A} m_{D \rightarrow B}(x_B)$$

*Proof.* The proof follows from applying Theorem 4.3.1 to the belief term in messages. □

## 4.4 Conditional Beliefs and Cutset Coding

One of the steps of cutset coding is conditioning on the cutset. Recall that in the Ising case, the cutset was simply the boundary and conditioning amounted to “collapsing” the “energy” of the cutset onto a relevant strip via some edge potentials. With complex cliques, the conditioning process is a bit more complicated. For example, consider the case where a clique simultaneously contains sites from two distinct clusters, and also the cutset (see Figure 4.2).

### 4.4.1 Conditional Beliefs

To encode a cluster  $A$  conditioned on cutset  $L$ , we need to know  $P(X_A = x_A | X_L = x_L)$ .

**Definition 4.4.1.** *Conditional Belief*

Let  $L \subset S$  be some cluster to be conditioned on, and let  $x_L$  be the observed configuration on this cluster. Then for cluster  $A \subset S$ ,  $A \cap L = \emptyset$ , we define the **conditional belief** (or the belief of  $A$  conditioned on  $L$ ) to be  $Z_{A|L} = \{Z_{A|L}(x_A | x_L) : x_A \in \Lambda^A\}$ ,

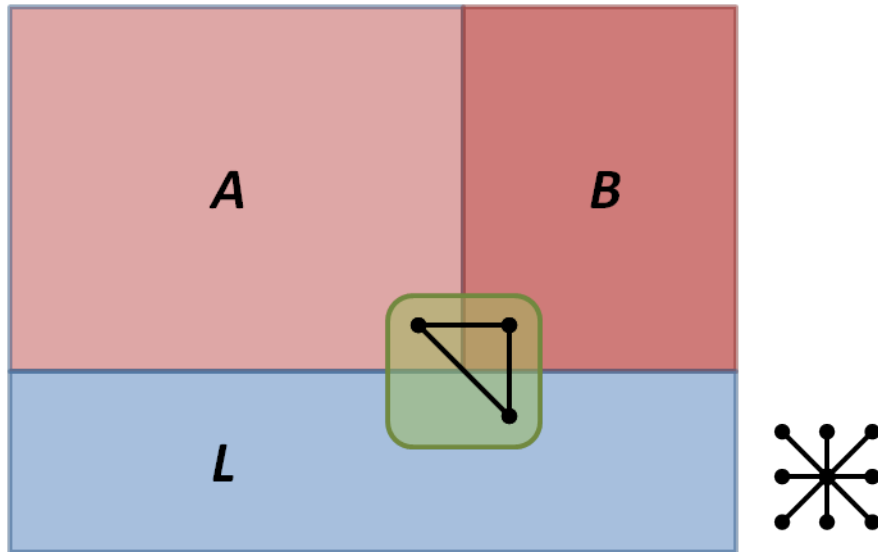


Figure 4.2: A cluster graph of 3 supernodes.  $A$  and  $B$  are nodes “to be encoded” and  $L$  is the cutset. An 8-point neighbourhood system is used here.

where,

$$Z_{A|L}(x_A|x_L) = \frac{1}{\Psi_L(x_L)} \sum_{x_{S \setminus (A \cup L)}} \prod_{C \in \mathcal{C}_S} \psi_C(x_A x_L x_{S \setminus (A \cup L)})$$

It is not difficult to see that  $Z_{A|L}(x_A|x_L) \propto P(X_A = x_A|X_L = x_L)$  since,

$$\begin{aligned}
P(x_A|x_L) &= \frac{P(x_A, x_L)}{P(x_L)} \\
&= \frac{Z_{AL}(x_A x_L)}{Z_L(x_L)} \\
&= \frac{\sum_{x_{S \setminus (A \cup L)}} \prod_{C \in \mathcal{C}_S} \psi_C(x_A x_L x_{S \setminus (A \cup L)})}{\sum_{x_{S \setminus L}} \prod_{C \in \mathcal{C}_S} \psi_C(x_L x_{S \setminus L})} \\
&= \frac{\Psi_L(x_L) \sum_{x_{S \setminus (A \cup L)}} \prod_{C \in \mathcal{C}_S} \psi_C(x_A x_L x_{S \setminus (A \cup L)})}{\Psi_L(x_L) \sum_{x_L} \sum_{x_{S \setminus (A \cup L)}} \prod_{C \in \mathcal{C}_S} \psi_C(x_L x_L x_{S \setminus (A \cup L)})} \\
&= \frac{Z_{A|L}(x_A|x_L)}{\sum_{x_L} Z_{A|L}(x_A|x_L)}
\end{aligned}$$

Now, consider the case of Figure 4.2 above where the graph is partitioned by clusters  $A$ ,  $B$ , and  $L$ . Let these clusters form the cluster graph  $\mathcal{G} = (\mathcal{A}, \Delta_{\mathcal{A}})$ , that is  $\mathcal{A} = \{A, B, L\}$ . What we should immediately note is that  $\mathcal{G}$  does not satisfy the minimum-radius condition,

$$\mathcal{C}_S \supset \mathcal{C}_{\mathcal{G}}$$

In fact, by inspection, we can write,

$$\begin{aligned}
\mathcal{C}_S &= \mathcal{C}_{\mathcal{G}} \cup \Delta_{A,B,L} \\
&= (\mathcal{C}_A \cup \mathcal{C}_B \cup \mathcal{C}_L) \cup (\Delta_{A,B} \cup \Delta_{A,L} \cup \Delta_{B,L}) \cup \Delta_{A,B,L}
\end{aligned}$$

Also if we consider only the sites in  $A$  and  $B$ , that is if we let  $S' = A \cup B$ , then the graph  $\mathcal{G}' = (\mathcal{A}', \Delta_{\mathcal{A}'})$ , where  $\mathcal{A}' = \{A, B\}$ , does indeed satisfy the minimum-radius



condition <sup>5</sup>. This is important to note because if we were interested in encoding  $\mathcal{G}'$  by itself as opposed to encoding it as a subgraph of  $\mathcal{G}$  conditioned on  $L$ , then the BP procedure detailed in Section 4.3 could be applied.

We now examine the following pathological examples (Figure 4.3).

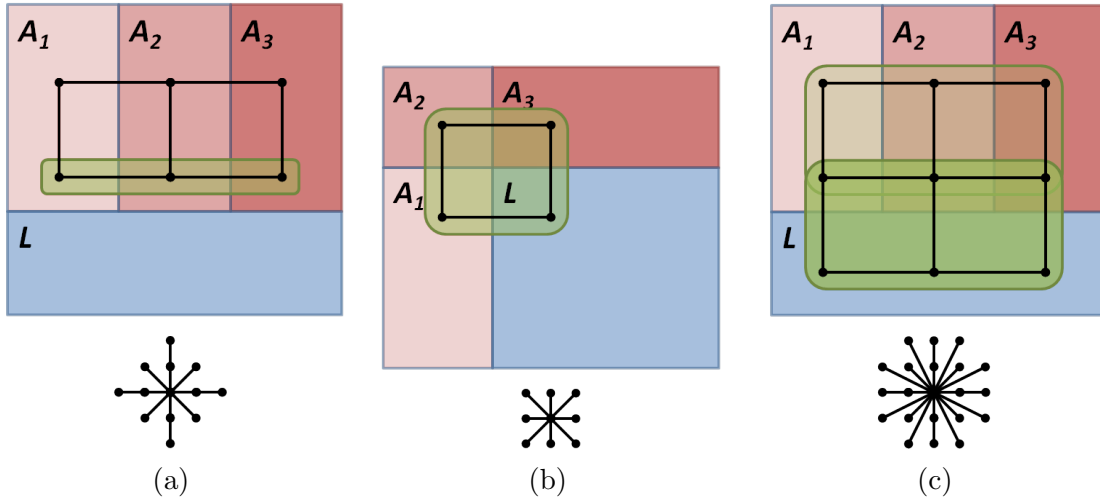


Figure 4.3: Various clique arrangements across 4 clusters with 12-point, 8-point, and 20-point neighbourhood systems.

Take the subgraph  $\mathcal{G}' = (\mathcal{A}', \Delta_{\mathcal{A}'})$  where  $\mathcal{A}' = \mathcal{A} \setminus L$ , then we notice a few potential problems. In (a), we see that  $\mathcal{A}'$  does not satisfy the minimum-radius condition, and so BP will not work <sup>6</sup>. In (b), we notice a particularly interesting situation where a clique exists across 4 different clusters. (c) simply illustrates that when using a larger neighbourhood system, the example in (a) actually exhibits the same phenomenon shown in (b).

Based on how  $\mathcal{G}'$  and  $L$  are chosen, the set of cliques not accounted for by  $\mathcal{C}_G$  can be quite diverse. <sup>7</sup> Fortunately, the following lemma tells us that these pathological

<sup>5</sup>Proven in Corollary 4.4.1 below.

<sup>6</sup>By Lemma 4.2.1, not satisfying the minimum-radius condition means  $\mathcal{G}'$  is also acyclic.

<sup>7</sup>With larger neighbourhoods, if multiple clusters closely border each other, many cliques will have sites in numerous clusters.

cliques can be avoided by choosing clusters that allow us to systematically keep track of all cliques with confidence.

Also, from now on, when we say that a cluster graph  $\mathcal{G} = (\mathcal{A}, \Delta_{\mathcal{A}})$  is **composed** of the subgraph  $\mathcal{G}'$  and some cluster  $L$ , we mean that,

$$\mathcal{A} = \mathcal{A}' \cup \{L\}$$

**Lemma 4.4.1.** *Unaccounted Cliques Involve at Most 3 Clusters*

*Let  $\mathcal{G}$  be a cluster graph composed of the subgraph  $\mathcal{G}'$ , and the cutset cluster  $L$ . If  $\mathcal{G}'$  satisfies the minimum-radius condition, then  $\mathcal{C}_S \setminus \mathcal{C}_{\mathcal{G}}$  is composed of cliques that involve exactly 3 different clusters including  $L$ .*

Essentially, this means that as long as we *pick*  $\mathcal{G}'$  in such a way as to ensure the minimum-radius condition, then we do not need to worry about the cases where cliques exist across more than 3 clusters. This is analogous to talking about the order of cliques - here, we can think of the *order of supernode interactions* being limited to at most 3.

*Proof.* Assume the Lemma is not true. Let  $\mathcal{G}'$  satisfy the minimum-radius condition, and without loss of generality, say  $\exists C \subset S$  such that  $C$  contains sites from 4 different clusters (one of which must be  $L$ ). Denote these clusters by  $A_i, A_j, A_k \in \mathcal{A}' \subset \mathcal{A}$ . Arbitrarily select sites  $a_i, a_j, a_k, a_l \in C$ , where  $a_i \in A_i, a_j \in A_j, a_k \in A_k, a_l \in L$ . By definition of cliques,  $a_i, a_j, a_k$ , and  $a_l$  are pairwise neighbours of each other, and so  $\{a_i, a_j, a_k\}$  is a valid clique. This means  $\Delta_{A_i, A_j, A_k}$  is non-empty. However, since  $\mathcal{G}'$  satisfies the minimum-radius condition,  $\Delta_{A_i, A_j, A_k}$  is necessarily empty, and so we have a contradiction.  $\square$

Note that it may be the case that  $\mathcal{G}$  itself satisfies the minimum-radius condition. In this case, it is obvious that  $\mathcal{C}_S \setminus \mathcal{C}_G = \emptyset$ .

**Corollary 4.4.1.** *Let some cluster graph be composed of the subgraph  $\mathcal{G}'$ , and cluster  $L$ . Let  $\Delta_{A,B} \in \Delta_{\mathcal{A}'}$  be a super-cut-edge in  $\mathcal{G}'$ . Then for the cluster graph  $\mathcal{G}$  composed of  $\mathcal{G}^* = (\{S'_A, S'_B\}, \Delta_{\{S'_A, S'_B\}})$  and  $L$ , we have,*

$$\begin{aligned} \mathcal{C}_S &= \left( \mathcal{C}_{S'_A} \cup \mathcal{C}_{S'_B} \cup \mathcal{C}_L \right) \cup \left( \Delta_{S'_A, S'_B} \cup \Delta_{S'_A, L} \cup \Delta_{S'_B, L} \right) \cup \Delta_{S'_A, S'_B, L} \\ &= \mathcal{C}_G \cup \Delta_{S'_A, S'_B, L} \end{aligned}$$

This is somewhat of an obvious result since  $S'_A \cup S'_B \cup L = S$  and so any *unaccounted* cliques can only involve at most those 3 nodes. For the sake of illustrating how Lemma 4.4.1 is useful, we present the following proof.

*Proof.* Since  $\Delta_{A,B} \in \Delta_{\mathcal{A}'}$  is a super-cut-edge on  $\mathcal{G}'$ , we know that  $S'_A \cup S'_B = S'$ . By Lemma 4.2.4 we have  $\mathcal{C}_{S'} = (\mathcal{C}_{S'_A} \cup \mathcal{C}_{S'_B}) \cup \Delta_{S'_A, S'_B}$ , meaning  $\mathcal{G}^*$  satisfies the minimum-radius condition, and so by Lemma 4.4.1, we know that any  $\mathcal{C}_S \setminus \mathcal{C}_G$  involve 3 clusters. In this case, these clusters are  $S'_A$ ,  $S'_B$ , and  $L$ .  $\square$

Going back to the initial example in Figure 4.2. Since  $\mathcal{G}' = (\{A, B\}, \Delta_{\{A, B\}})$  satisfies the minimum-radius condition, Corollary 4.4.1 confirms the observation that,

$$\begin{aligned} \mathcal{C}_S &= \left( \mathcal{C}_A \cup \mathcal{C}_B \cup \mathcal{C}_L \right) \cup \left( \Delta_{A,B} \cup \Delta_{A,L} \cup \Delta_{B,L} \right) \cup \Delta_{A,B,L} \\ &= \mathcal{C}_G \cup \Delta_{A,B,L} \end{aligned}$$

Using this, we can get the following expression for the conditional belief of  $A$  on

$L$ ,

$$\begin{aligned}
Z_{A|L}(x_A|x_L) &= \frac{1}{\Psi_L(x_L)} \sum_{x_{S \setminus (A \cup L)}} \prod_{C \in \mathcal{C}_S} \psi_C(x_A x_L x_{S \setminus (A \cup L)}) \\
&= \frac{1}{\Psi_L(x_L)} \sum_{x_B} \Psi_A(x_A) \Psi_B(x_B) \Psi_L(x_L) \Psi_{A,B}(x_A x_B) \Psi_{A,L}(x_A x_L) \Psi_{B,L}(x_B x_L) \\
&\quad \Psi_{A,B,L}(x_A x_B x_L) \\
&= \left( \Psi_A(x_A) \Psi_{A,L}(x_A x_L) \right) \sum_{x_B} \Psi_{A,B}(x_A x_B) \Psi_{A,B,L}(x_A x_B x_L) \\
&\quad \left( \Psi_B(x_B) \Psi_{B,L}(x_B x_L) \right),
\end{aligned}$$

where the first step is by Corollary 4.2.1. It is no surprise that this look strangely similar to the BP algorithm. Before we continue to formally define Conditional Belief Propagation, we define the following,

**Definition 4.4.2. *Conditioned Supernode Self-potential***

Let  $\mathcal{G}$  be a cluster graph. For neighbouring supernodes  $A, L \in \mathcal{A}$ , we define the supernode self-potential of  $A$  conditioned on  $L$  to be,

$$\Psi_{A|L}(x_A x_L) = \Psi_A(x_A) \Psi_{A,L}(x_A x_L)$$

Note that if  $A$  and  $L$  are not neighbours, then the result is equivalent to just the self-potential of supernode  $A$ .

**Definition 4.4.3. *Conditioned Superedge potential***

Let  $\mathcal{G}$  a cluster graph. For pairwise-neighbouring supernodes  $A, B, L \in \mathcal{A}$ , we define

the superedge potential of  $\Delta_{A,B} \in \Delta_A$  conditioned on  $L$  to be,

$$\Psi_{A,B|L}(x_A x_B x_L) = \Psi_{A,B}(x_A x_B) \Psi_{A,B,L}(x_A x_B x_L)$$

## 4.4.2 Conditional Belief Propagation

**Definition 4.4.4.** *Conditional Belief on Cluster subgraphs*

Let  $\mathcal{G}$  be a cluster graph composed of a cluster subgraph  $\mathcal{G}'$ , and cutset  $L$ . Let  $x_L$  be the configuration on the observed cluster  $L$ . Let  $\mathcal{G}'_{A \setminus B} = (\mathcal{A}'_A, \Delta_{\mathcal{A}'_A})$  be the cluster subgraph containing cluster  $A$  after removing  $\Delta_{A,B} \in \Delta_{\mathcal{A}'}$  (a subgraph of  $\mathcal{G}'$ ). Then the **conditional subgraph belief** of cluster  $A$  on the subgraph  $\mathcal{G}'_{A \setminus B}$  is defined to be  $Z_{A \setminus B|L} = \{Z_{A \setminus B|L}(x_A) : x_A \in \Lambda^A\}$  where,

$$\begin{aligned} Z_{A \setminus B|L}(x_A | x_L) &= \sum_{x_{S'_A \setminus A}} \prod_{C \in (\mathcal{C}_{S'_A} \cup \Delta_{S'_A, L})} \psi_C(x_A x_{S'_A \setminus A} x_L) \\ &= \sum_{x_{S'_A \setminus A}} \Psi_{S'_A}(x_A x_{S'_A \setminus A}) \Psi_{S'_A, L}(x_A x_{S'_A \setminus A} x_L) \end{aligned}$$

We now define conditional messages.

**Definition 4.4.5.** *Conditional Messages for Cluster Graphs*

Let  $\mathcal{G}$  be a cluster graph composed of a cluster subgraph  $\mathcal{G}'$ , and cutset  $L$ . Let  $x_L$  be the configuration on the observed cluster  $L$ . For some super-cut-edge  $\Delta_{A,B} \in \Delta_{\mathcal{A}'}$  (a super-cut-edge of  $\mathcal{G}'$ ), the **conditional message** from supernode  $B$  to supernode  $A$  is defined to be  $m_{B \rightarrow A|L} = \{m_{B \rightarrow A|L}(x_A | x_L) : x_A \in \Lambda^A\}$  where,

$$m_{B \rightarrow A|L}(x_A | x_L) = \sum_{x_B} \Psi_{A,B}(x_A x_B) \Psi_{A,B,L}(x_A x_B x_L) Z_{B \setminus A|L}(x_B | x_L)$$

**Lemma 4.4.2.** *Conditional Cluster Subgraph Message Passing*

Let  $\mathcal{G}$  be a cluster graph composed of cluster subgraph  $\mathcal{G}'$  and cutset  $L$ . Let  $x_L$  be the configuration on the observed cluster  $L$ . Let  $\Delta_{A,B} \in \Delta_{\mathcal{A}'}$  be a super-cut-edge of  $\mathcal{G}'$ , then,

$$Z_{A|L}(x_A|x_L) = Z_{A \setminus B|L}(x_A|x_L)m_{B \rightarrow A|L}(x_A|x_L)$$

The proof is analogous to the proof for Lemma 4.3.1.

*Proof.* By Corollary 4.4.1, we know that,

$$\mathcal{C}_S = \left( \mathcal{C}_{S'_A} \cup \mathcal{C}_{S'_B} \cup \mathcal{C}_L \right) \cup \left( \Delta_{S'_A, S'_B} \cup \Delta_{S'_A, L} \cup \Delta_{S'_B, L} \right) \cup \Delta_{S'_A, S'_B, L}.$$

Using the approach we took with the example given in Figure 4.2, we get the following breakdown of cliques.

$$\begin{aligned} Z_{A|L}(x_A|x_L) &= \frac{1}{\Psi_L(x_L)} \sum_{x_{S \setminus (A \cup L)}} \prod_{C \in \mathcal{C}_S} \psi_C(x_A x_L x_{S \setminus (A \cup L)}) \\ &= \sum_{x_{S \setminus (A \cup L)}} \left( \Psi_{S'_A}(x_A x_{S \setminus (A \cup L)}) \Psi_{S'_A, L}(x_A x_L x_{S \setminus (A \cup L)}) \right) \\ &\quad \left( \Psi_{S'_A, S'_B}(x_A x_{S \setminus (A \cup L)}) \Psi_{S'_A, S'_B, L}(x_A x_L x_{S \setminus (A \cup L)}) \right) \\ &\quad \left( \Psi_{S'_B}(x_{S \setminus (A \cup L)}) \Psi_{S'_B, L}(x_L x_{S \setminus (A \cup L)}) \right). \end{aligned}$$

Since  $S \setminus (A \cup L) = S'_B \cup (S'_A \setminus A)$ , we can break down the sum.

$$Z_{A|L}(x_A|x_L) = \sum_{x_{S'_A \setminus A}} \left( \Psi_{S'_A}(x_A x_{S'_A \setminus A}) \Psi_{S'_A, L}(x_A x_{S'_A \setminus A} x_L) \right) \\ \sum_{x_{S'_B}} \left( \Psi_{S'_A, S'_B}(x_A x_{S'_A \setminus A} x_{S'_B}) \Psi_{S'_A, S'_B, L}(x_A x_{S'_A \setminus A} x_{S'_B} x_L) \right) \\ \Psi_{S'_B}(x_{S'_B}) \Psi_{S'_B, L}(x_{S'_B} x_L) \Big),$$

where the sums were separated based on the dependency of the potential functions.

Since  $S'_B = B \cup (S'_B \setminus B)$ , we can again break the sum into a double sum.

$$Z_{A|L}(x_A|x_L) = \sum_{x_{S'_A \setminus A}} \left( \Psi_{S'_A}(x_A x_{S'_A \setminus A}) \Psi_{S'_A, L}(x_A x_{S'_A \setminus A} x_L) \right) \\ \sum_{x_B} \sum_{x_{S'_B \setminus B}} \left( \Psi_{S'_A, S'_B}(x_A x_{S'_A \setminus A} x_{S'_B}) \Psi_{S'_A, S'_B, L}(x_A x_{S'_A \setminus A} x_{S'_B} x_L) \right) \\ \Psi_{S'_B}(x_{S'_B}) \Psi_{S'_B, L}(x_{S'_B} x_L) \Big).$$

Since  $\mathcal{G}'_{A \setminus B}$  and  $\mathcal{G}'_{B \setminus A}$  were induced by removing the super-cut-edge  $\Delta_{A, B} \in \Delta_{\mathcal{A}'}$ , we have that,

$$\Delta_{S'_A, S'_B} = \Delta_{A, B}$$

$$\Delta_{S'_A, S'_B, L} = \Delta_{A, B, L},$$

and so the expression simplifies to,

$$Z_{A|L}(x_A|x_L) = \sum_{x_{S'_A \setminus A}} \left( \Psi_{S'_A}(x_A x_{S'_A \setminus A}) \Psi_{S'_A, L}(x_A x_{S'_A \setminus A} x_L) \right) \\ \sum_{x_B} \left( \Psi_{A, B}(x_A x_B) \Psi_{A, B, L}(x_A x_B x_L) \right) \\ \sum_{x_{S'_B \setminus B}} \Psi_{S'_B}(x_B x_{S'_B \setminus B}) \Psi_{S'_B, L}(x_B x_{S'_B \setminus B} x_L) \Big),$$

where again, the sums are separated based on the dependency of potential functions. Noticing that the first and last term are simply conditional beliefs on subgraphs, we get,

$$Z_{A|L}(x_A|x_L) = Z_{A \setminus B|L}(x_A|x_L) \sum_{x_B} \Psi_{A, B}(x_A x_B) \Psi_{A, B, L}(x_A x_B x_L) Z_{B \setminus A|L}(x_B|x_L) \\ = Z_{A \setminus B|L}(x_A|x_L) m_{B \rightarrow A|L}(x_A|x_L).$$

□

Lemma 4.4.2 provides the basis for Conditional BP on a cutset  $L$ . Assuming that  $\mathcal{G}'$  is an acyclic graph, then it satisfies the minimum-radius condition, and every subgraph satisfies it as well. Also, since every superedge is a super-cut-edge, we can decompose the conditional beliefs.

**Theorem 4.4.1.** *Conditional Belief Decomposition for Acyclic Cluster Graphs*

Let  $\mathcal{G}$  be a cluster graph composed of acyclic cluster subgraph  $\mathcal{G}'$  and cutset  $L$ . Taking



cluster  $A \in \mathcal{A}'$ , we have,

$$Z_{A|L}(x_A|x_L) = \Psi_{A|L}(x_A x_L) \prod_{B \in \gamma A} m_{B \rightarrow A|L}(x_A|x_L).$$

*Proof.* The proof is analogous to the proof for Theorem 4.3.1. Since  $\mathcal{G}'$  is acyclic, we can repeatedly apply Lemma 4.4.2 to subgraphs *conditioned* on  $L$ <sup>8</sup>.  $\square$

When comparing this procedure to the simpler one in the Ising case, this makes heuristic sense. The conditional beliefs do indeed show a “collapsing” of potentials from the cutset to the cluster graph of interest. At the same time, because of the higher order cliques, the superedges require some “conditioning” as well. We view these superedges as having some “ends” that hold constant (observed) value. Finally, Lemma 4.4.1 and Corollary 4.4.1 assures us that conditioning will never result in one of the pathological cases we discussed previously where some cliques are unaccounted for.

**Theorem 4.4.2.** *Conditional Message Recursion on Cluster Graphs*

Let  $\mathcal{G}$  be a cluster graph composed of acyclic cluster subgraph  $\mathcal{G}'$  and cutset  $L$ . Messages can be expressed recursively by,

$$m_{B \rightarrow A|L}(x_A|x_L) = \sum_{x_B} \Psi_{A,B}(x_A x_B) \Psi_{A,B,L}(x_A x_B x_L) \Psi_{B|L}(x_B|x_L) \prod_{D \in (\gamma B) \setminus A} m_{D \rightarrow B|L}(x_B|x_L)$$

---

<sup>8</sup>The cluster subgraphs at each iteration are composed of a subgraph induced by removing a super-cut-edge and  $L$ .

## 4.5 Classifying Cliques and Potentials

The following classification for cliques and potentials will come in useful when discussing estimation for MRF's with complex cliques.

### 4.5.1 Clique Types

**Definition 4.5.1.** *Geometrically Related Cliques*

If two cliques are related to each other by some constant shift in sites, then they are **geometrically related**. That is, assume that sites are identified by the vector  $\vec{s} = \langle x, y \rangle$ . Let  $C = \{\vec{s}_1, \dots, \vec{s}_n\}$ , and  $C^* = \{\vec{s}_1^*, \dots, \vec{s}_n^*\}$ , then  $\vec{s}_i = \vec{s}_i^* + \vec{d} \forall 1 \leq i \leq n$  where  $\vec{d}$  is some constant vector in  $\mathbb{R}^2$ .

**Definition 4.5.2.** *Clique Type*

A **clique type** refers to a class of geometrically related cliques. Given a neighbourhood system  $N$ , we enumerate all possible clique types and denote the set of these types by  $\mathcal{C} = \{C_i\}$ . For  $A \subseteq S$ ,  $C_i(A)$  is the set of all cliques of type  $i$  on  $A$ .

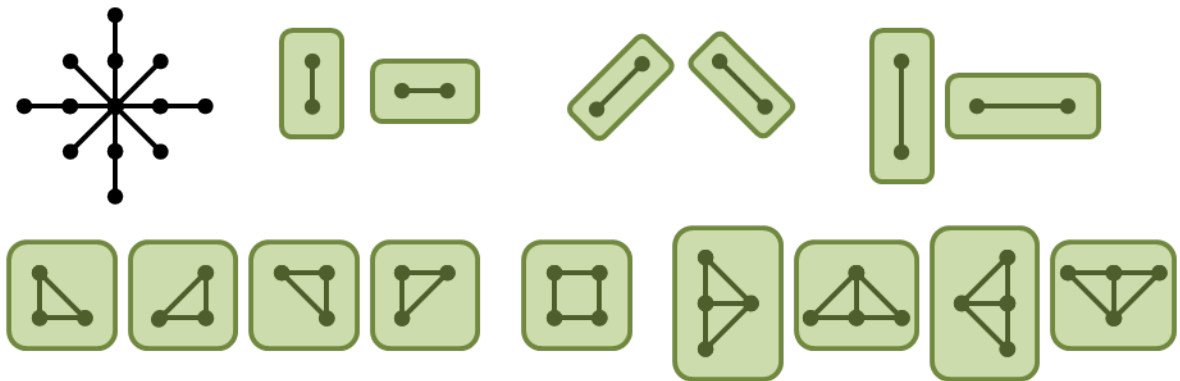


Figure 4.4: Clique types for a 12-point neighbourhood system.

Figure 4.4 shows possible clique types for different neighbourhood systems. Note

how it is possible for a neighbourhood to contain more than one clique of a certain type - this is generally true for larger neighbourhoods, and cliques of lower order.

**Lemma 4.5.1.** *Clique Types and Total Cliques*

Let  $\mathcal{M} = (S, N, \mathcal{V})$  be an MRF with clique type  $\mathfrak{C}$ . Then for  $A \subseteq S$ ,

$$\mathcal{C}_A = \bigcup_{i=1}^{|\mathfrak{C}|} \mathbb{C}_i(A)$$

*Proof.* Result follows by definition of clique type. □

**Corollary 4.5.1.** *Clique Types and Manipulating Potentials*

Let  $\mathcal{M} = (S, N, \mathcal{V})$  be an MRF with clique type  $\mathfrak{C}$ . Then for  $A \subseteq S$ ,

$$\Psi_A(x_A) = \exp \left( - \sum_{i=1}^{|\mathfrak{C}|} \sum_{C \in \mathbb{C}_i(A)} V_C(x_A) \right)$$

*Proof.* By Lemma 4.5.1

$$\begin{aligned} \Psi_A(x_A) &= \prod_{C \in \mathcal{C}_A} \psi_C(x_A) \\ &= \prod_{i=1}^{|\mathfrak{C}|} \left( \prod_{C \in \mathbb{C}_i} \psi_C(x_A) \right) \\ &= \exp \left( - \sum_{i=1}^{|\mathfrak{C}|} \sum_{C \in \mathbb{C}_i(A)} V_C(x_A) \right) \end{aligned}$$

□

## 4.5.2 Linear Parameter Potentials

As discussed previously, potential functions are often governed by a parameter  $\theta$ . Here, we are interested in a particular kind of potential functions we call the **linear parameter potentials**.

**Definition 4.5.3.** *Linear Parameter Potentials*

A potential function  $V_C$  is a linear parameter potential if it is of the following form,

$$V_C(x_C) = \xi_C(x_C)\theta_C + c$$

where  $\theta_C$  is the governing parameter for the potential.  $\xi_C$  is some real valued function mapping from the configuration on clique  $C$ ; and  $c \in \mathbb{R}$  is some constant (call these the **ranking function** and **ranking constant**).

We denote  $\Theta$  the space of parameters that govern the potentials for a MRF. In the case of the Ising model,  $\theta = \{\theta_1, \theta_2\}$ ,  $\theta \in \Theta$ . The following definitions follow.

**Definition 4.5.4.** *Clique Type Parameters*

For a MRF with  $|\mathfrak{C}|$  enumerated clique types in  $\mathfrak{C}$ , the set  $\theta = \{\theta_1, \dots, \theta_{|\mathfrak{C}|}\}$  will denote the the corresponding parameters for each clique type.

**Definition 4.5.5.** *Clique Type Potentials*

For a MRF with  $|\mathfrak{C}|$  enumerated clique types in  $\mathfrak{C}$ , the set  $\mathcal{V} = \{V_1, \dots, V_{|\mathfrak{C}|}\}$  will denote the the corresponding potential functions for each clique type.

Likewise,  $\xi = \{\xi_i\}$  and  $c = \{c_i\}$  denote the associated ranking functions and constants on relevant sites. We shall soon see that MRFS with only linear parameter potentials can be estimated by LS or MLE.

## 4.6 Estimation

We now go over LS and ML estimation techniques for MRF's with complex cliques. As it turns out, as long as the MRF has only linear parameter potentials, the estimation procedure does not change much.

### 4.6.1 Least Squares

**Lemma 4.6.1.** *Let MRF  $\mathcal{M} = (S, N, \mathcal{V})$  have only linear parameter potentials, then for some site  $s \in S$  and  $\lambda_s, \lambda_s^* \in \Lambda$ ,*

$$\ln \left( \frac{P(X_s = \lambda_s | X_{N_s} = x_{N_s})}{P(X_s = \lambda_s^* | X_{N_s} = x_{N_s})} \right) = - \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathcal{C}_k(s \cup N_s)} \left( \theta_k(\xi_k(x_C) - \xi_k(x_C)) \right)$$

For all intents and purposes, the probabilities above do not need to be conditional; we use them to illustrate that during LS estimation, we hold the configuration of the neighbourhood constant.

*Proof.*

$$\begin{aligned} \frac{P(X_s = \lambda_s | X_{N_s} = x_{N_s})}{P(X_s = \lambda_s^* | X_{N_s} = x_{N_s})} &= \frac{P(X_s = \lambda_s, X_{N_s} = x_{N_s})}{P(X_s = \lambda_s^*, X_{N_s} = x_{N_s})} \\ &= \frac{Z_{(s \cup N_s)}(\lambda_s x_{N_s})}{Z_{(s \cup N_s)}(\lambda_s^* x_{N_s})} \\ &= \frac{\sum_{x_{S \setminus (s \cup N_s)}} \prod_{C \subset S} \psi_C(\lambda_s x_{N_s} x_{S \setminus (s \cup N_s)})}{\sum_{x_{S \setminus (s \cup N_s)}} \prod_{C \subset S} \psi_C(\lambda_s^* x_{N_s} x_{S \setminus (s \cup N_s)})}. \end{aligned}$$

By Corollary 4.2.1, we have,

$$\begin{aligned}
\frac{Z_{(s \cup N_s)}(\lambda_s x_{N_s})}{Z_{(s \cup N_s)}(\lambda_s^* x_{N_s})} &= \frac{\Psi_{(s \cup N_s)}(\lambda_s x_{N_s}) \sum_{x_{S \setminus (s \cup N_s)}} \Psi_{S \setminus (s \cup N_s)}(x_{S \setminus (s \cup N_s)})}{\Psi_{(s \cup N_s)}(\lambda_s^* x_{N_s}) \sum_{x_{S \setminus (s \cup N_s)}} \Psi_{S \setminus (s \cup N_s)}(x_{S \setminus (s \cup N_s)})} \\
&= \frac{\Psi_{(s \cup N_s), (S \setminus (s \cup N_s))}((\lambda_s x_{N_s} x_{S \setminus (s \cup N_s)})}{\Psi_{(s \cup N_s), (S \setminus (s \cup N_s))}((\lambda_s^* x_{N_s} x_{S \setminus (s \cup N_s)})} \\
&= \frac{\Psi_{(s \cup N_s)}(\lambda_s x_{N_s})}{\Psi_{(s \cup N_s)}(\lambda_s^* x_{N_s})} \\
&= \frac{\exp\left(-\sum_{C \subset (s \cup N_s)} V_C(\lambda_s x_{N_s})\right)}{\exp\left(-\sum_{C \subset (s \cup N_s)} V_C(\lambda_s^* x_{N_s})\right)} \\
&= \exp\left(-\sum_{C \subset (s \cup N_s)} \left(V_C(\lambda_s x_{N_s}) - V_C(\lambda_s^* x_{N_s})\right)\right)
\end{aligned}$$

.

By Corollary 4.5.1, and by the fact that all the potentials are of the linear parameter type, we have,

$$-\sum_{C \subset (s \cup N_s)} \left(\theta_C(\xi(\lambda_s x_{N_s}) - \xi(\lambda_s^* x_{N_s}))\right) = -\sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathcal{C}_k(s \cup N_s)} \left(\theta_k(\xi_k(\lambda_s x_{N_s}) - \xi_k(\lambda_s^* x_{N_s}))\right)$$

and so,

$$\ln\left(\frac{P(X_s = \lambda_s | X_{N_s} = x_{N_s})}{P(X_s = \lambda_s^* | X_{N_s} = x_{N_s})}\right) = -\sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathcal{C}_k(s \cup N_s)} \left(\theta_k(\xi_k(\lambda_s x_{N_s}) - \xi_k(\lambda_s^* x_{N_s}))\right).$$

□

$P(X_{s \cup N_s} = \lambda_s x_{N_s})$  and  $P(X_{s \cup N_s} = \lambda_s^* x_{N_s})$  can be empirically determined by collecting data from a training set. Once these are found, we end up with a set of overdetermined linear equations which we could use to solve for  $\theta \in \Theta$ . To be precise, there are,

$$\binom{r}{2} r^{|\partial s|}$$

equations, where  $r = |\Lambda|$  is the cardinality of the phase space (number of possible values that a site can take). There are  $r^{|\partial s|}$  possible neighbourhood configurations around the site  $s$ , and for each specific neighbourhood, there are  $\binom{r}{2}$  possible combinations for comparing differing values on site  $s$ .

The following steps summarize the estimation procedure.

1. Collect data for various neighbourhood and site configurations.
2. Construct an over-determined system of equations with  $\theta = \{\theta_1, \dots, \theta_{|\mathfrak{C}|}\}$  unknowns.
  - (a) Use data from (1) to compute the L.H.S. of Lemma 4.6.1.
  - (b) Compute the relevant coefficients for each  $\theta_k$  using the R.H.S of Lemma 4.6.1.
3. Solve by least squares method.

### 4.6.2 MLE

As discussed previously, ML estimation attempts to maximize the *likelihood*,  $L(\theta)$ , over the space of parameters  $\Theta$ . The likelihood function is the probability of some

observations given some parameter  $\theta \in \Theta$ ,

$$\begin{aligned} L(\theta) &= f(x_1, \dots, x_n | \theta) \\ &= f(x_1 | \theta) \dots f(x_n | \theta) \\ &= \prod_{i=1}^n f(x_i | \theta). \end{aligned}$$

When appropriate, this is equivalent to maximizing the *log likelihood*  $\ln L(\theta)$ ,

$$\ln(L(\theta)) = \sum_{i=1}^n \ln(f(x_i | \theta)).$$

In our case, it is obvious that the MRF is part of the exponential family by the way it is defined, and so,

$$\begin{aligned} \hat{\theta} &= \arg \max_{\theta \in \Theta} L(\theta) \\ &= \arg \max_{\theta \in \Theta} \ln(L(\theta)). \end{aligned}$$

More specifically, we have,

$$f_{X|\Theta}(x|\theta) = \frac{1}{Q(\theta)} \prod_{C \subset V} \exp(-V_C(x, \theta)),$$

where the updated notation of  $V_C$  appropriately indicates dependency on  $\theta$ . Hence,

$$L(\theta) = \frac{1}{Q(\theta)^n} \prod_{i=1}^n \left( \exp \left( - \sum_{C \subset V} V_C(x_i, \theta) \right) \right).$$



Maximizing the log likelihood over  $\Theta$  then becomes,

$$\hat{\theta} = \arg \max_{\theta \in \Theta} \ln (L(\theta)) = \arg \max_{\theta \in \Theta} \left( -n \ln Q(\theta) - \sum_{i=1}^n \left( \sum_{C \subset V} V_C(x_i, \theta) \right) \right).$$

As we discussed previously, direct computation of the partition function  $Q(\theta)$  is intractable, and so  $\hat{\theta}$  can only be approximated by locally searching via gradient ascent. To do this, we need to compute the gradient at each step.

$$\left\{ \frac{\partial}{\partial \theta_k} \ln (L(\theta)) \right\}$$

We present the following Lemma regarding the gradient for MRF's with linear parameter potentials.

**Lemma 4.6.2.** *Let MRF  $\mathcal{M} = (S, N, \mathcal{V})$  be a MRF which has only linear parameter potentials, then,*

$$\frac{\partial}{\partial \theta_k} \ln (L(\theta)) = n \left( \mu(t_k) - t_k(x_1, \dots, x_n) \right)$$

where  $\mathbb{C}_k \in \mathfrak{C}$  and,

$$t_k(x_1, \dots, x_n) = \frac{1}{n} \sum_{i=1}^n \sum_{C \in \mathbb{C}_k(S)} \xi_k(x_{i_C})$$

$$\mu(t_k) = \sum_{x \in \Lambda^S} \pi(x) \left( \sum_{C \in \mathbb{C}_k(S)} \xi_k(x_{i_C}) \right)$$

are **statistics** with respect to  $\theta_k \in \theta$ .

*Proof.* First, note that by Corollary 4.5.1 we have,

$$\begin{aligned} \ln(L(\theta)) &= -n \ln Q(\theta) - \sum_{i=1}^n \left( \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathbb{C}_k(S)} V_C(x_i, \theta) \right) \\ &= -n \ln Q(\theta) - \sum_{i=1}^n \left( \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathbb{C}_k(S)} (\theta_k \xi_k(x_{i_C}) + c_k) \right) \end{aligned}$$

The last part follows from the fact that MRF has only linear parameter potentials. The  $x_{i_C}$  refers to the configuration on clique  $C$  for sample  $i$ ; recall that potentials depend only on its relevant clique. Now, given the statistics  $\{t_k\}$ , we get,

$$\begin{aligned} \ln(L(\theta)) &= -n \ln Q(\theta) - \sum_{i=1}^n \left( \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathbb{C}_k(S)} (\theta_k \xi_k(x_{i_C}) + c_k) \right) \\ &= -n \ln Q(\theta) - n \left( \sum_{k=1}^{|\mathcal{C}|} \theta_k \left( \frac{1}{n} \sum_{i=1}^n \sum_{C \in \mathbb{C}_k(S)} \xi_k(x_{i_C}) \right) \right) \\ &\quad - \sum_{i=1}^n \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathbb{C}_k(S)} c_k \\ &= -n \ln Q(\theta) - n \left( \sum_{k=1}^{|\mathcal{C}|} \theta_k t_k(x_1, \dots, x_n) \right) - \sum_{i=1}^n \sum_{k=1}^{|\mathcal{C}|} c_k |\mathbb{C}_k(S)|. \end{aligned}$$

Now, taking the partial derivative with respect to  $\theta_k$ , we have,

$$\frac{\partial}{\partial \theta_k} \ln(L(\theta)) = -n \frac{1}{Q(\theta)} \frac{\partial}{\partial \theta_k} Q(\theta) - n t_k(x_1, \dots, x_n).$$

Taking a closer look at the partition function  $Q(\theta)$ ,

$$\begin{aligned} Q(\theta) &= \sum_{x \in \Lambda^S} \exp \left( - \sum_{C \subset S} V_C(x, \theta) \right) \\ &= \sum_{x \in \Lambda^S} \exp \left( - \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathcal{C}_k(S)} \left( \theta_k \xi_k(x_C) + c_k \right) \right) \end{aligned}$$

$$\frac{\partial}{\partial \theta_k} Q(\theta) = \sum_{x \in \Lambda^S} \exp \left( - \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathcal{C}_k(S)} \left( \theta_k \xi_k(x_C) + c_k \right) \right) \left( - \sum_{C \in \mathcal{C}_k(S)} \xi_k(x_C) \right).$$

Thus we obtain the following expression,

$$\begin{aligned} \frac{\partial}{\partial \theta_k} \ln(L(\theta)) &= -n \frac{1}{Q(\theta)} \frac{\partial}{\partial \theta_k} Q(\theta) - nt_k(x_1, \dots, x_n) \\ &= -n \sum_{x \in \Lambda^S} \frac{1}{Q(\theta)} \exp \left( - \sum_{k=1}^{|\mathcal{C}|} \sum_{C \in \mathcal{C}_k(S)} \theta_k \xi_k(x_C) \right) \left( - \sum_{C \in \mathcal{C}_k(S)} \xi_k(x_C) \right) \\ &\quad - nt_k(x_1, \dots, x_n) \\ &= n \sum_{x \in \Lambda^S} \pi(x) \left( \sum_{C \in \mathcal{C}_k(S)} \xi_k(x_{i_C}) \right) - nt_k(x_1, \dots, x_n) \\ &= n \left( \mu(t_k) - t_k(x_1, \dots, x_n) \right). \end{aligned}$$

□

As mentioned in Chapter 2, calculating the gradient amounts to moment matching to a set of training data. We now review the estimation procedure when approximating  $\hat{\theta}$  by gradient ascent.

1. Approximate  $\{\mu(t_k)\}$  by taking a “sufficiently large” set of training data and

computing the relevant statistics  $\{t_k\}$ .

2. Start by picking an initial guess  $\theta = \{\theta_k\}$ .
3. Generate a “sufficiently large” set of  $n$  samples  $(x_1, \dots, x_n)$  with the Gibbs Sampler using  $\theta$  as parameters, and compute the relevant statistics  $\{t_k(x_1, \dots, x_n)\}$  over the set of samples.
4. Using Lemma 4.6.2, compute the gradient using  $\{\mu(t_k)\}$  from (1), and  $\{t_k(x_1, \dots, x_n)\}$  from (3).
5. Increment  $\theta$  in the direction of gradient and use this as the new “guess”. Again, note that we are maximizing  $L$  and hence need to travel in the direction of steepest ascent.
6. Repeat steps (3) to (5) until the gradient becomes small.

The maximum likelihood (ML) method is computationally time consuming since the Gibbs Sampler is used at every step to generate  $n$  samples. The number of samples needed for approximation increases exponentially with the number of sites  $|S|$ , and the size of phase space  $|\Lambda|$ . The time to generate a sample increases with  $|S|$  and  $|\Lambda|$  as well. Finally, with complex cliques and larger neighbourhood systems, the number of clique types  $|\mathfrak{C}|$  increases rapidly as well, increasing the number of statistics to be computed.

# Chapter 5

## Precoding with Complex Cliques

We now utilize the precoding procedure described in Chapter 3 with higher order cliques. The general procedure remains the same. The only adjustments we make are to steps (1) and (5),

- Step 1: Establish MRF and cutset specifications. *We now need to choose the appropriate cutset and clusters so to satisfy the **minimum radius condition**.*
- Step 5: Encode components. *We now use conditional belief propagation.*

Some results are presented here, and the Chapter ends with some suggestions for future work.

### 5.1 Minimum Radius Clustering

We saw from Chapter 4 that a prerequisite for the BP algorithm is a cluster graph  $\mathcal{G}$  composed of acyclic subgraph  $\mathcal{G}'$  and cutset  $L$ . In Chapter 3, the cutsets were horizontal strips 1-pixel in thickness. This segmented the image into multiple horizontal

slices of a certain column spacing. These slices were treated as individual MRF's and cluster graphs were generated by grouping sites within the same pixel column. This was then followed by arithmetic coding via BP. Since the neighbourhood system was of the 4-point Ising type, notice that for graph  $\mathcal{G}$  composed of a given slice and relevant cutsets<sup>1</sup>, the subgraph  $\mathcal{G}'$  was clearly acyclic since the “radius” of the neighbourhood is 1-pixel. That is, there were no cliques that extended beyond 1-pixel in either direction<sup>2</sup>, meaning all clique potentials within the graph were captured by the self potentials of supernodes and superedges. We now provide a more well defined statement of this method.

**Definition 5.1.1.** *Radius of a Neighbourhood*

Let the set of sites  $S$  be a finite 2-D grid identified by ordered pairs as per usual in  $\mathbb{R}^2$ . Then the **radius**, denoted  $r_s$ , of a neighbourhood  $N_s$ ,  $s \in S$ , is the largest distance between the specifying site  $s$  of  $N_s$  and any other site  $t \in N_s$ , where distance is as usually defined in  $\mathbb{R}^2$ . That is,

$$r_s = \max_{t \in N_s} d(s, t)$$

For the Ising 4-point neighbourhood,  $r_s = 1$ . Similarly, we define the *width* of a clique.

**Definition 5.1.2.** *Width of a Clique*

The **width** of a clique  $C$  is given by,

$$w_C = \max_{s, t \in C} d(s, t)$$

---

<sup>1</sup>Mainly the 1-pixel strips above and below “bordering” the slice.

<sup>2</sup>Towards the clusters immediately before or after. In this case the clusters immediately left and right.

It is not too hard to see that the width of a clique will always be less or equal to the radius of a neighbourhood.

**Lemma 5.1.1.** *Widths of Cliques Less or Equal to Radius of Neighbourhood*

*For some site  $s \in S$  and clique  $C \subset N_s$ ,*

$$w_C \leq r_s$$

*Proof.* The result follows by definition of cliques being a group of sites that are pairwise neighbours. By symmetry of neighbourhoods, the maximum width of a clique cannot exceed the radius of the neighbourhood.  $\square$

We now present the following result regarding cluster graphs which satisfy the minimum radius condition.

**Theorem 5.1.1.** *Minimum Radius Cluster Graphs*

*For a cluster graph  $\mathcal{G}' = (\mathcal{A}', \Delta_{\mathcal{A}'})$  induced by MRF  $\mathcal{M} = (S', N, \mathcal{V})$ ,  $\mathcal{G}'$  satisfies the minimum radius condition if for every cluster  $A_i \in \mathcal{A}'$ , we have that the following is true for any  $A_j, A_k \in \gamma A_i$ .*

$$d(a_j, a_k) > \max(r_{a_j}, r_{a_k})$$

$\forall a_j \in A_j$ , and  $a_k \in A_k$ .

That is to say, a cluster graph satisfies the minimum radius condition if given any cluster  $A_i$ , the minimum distance between any of its two neighbour-clusters  $A_j$  and  $A_k$  is greater than the radius of a neighbourhood.

*Proof.* Since the distance between any two sites of  $A_j$  and  $A_k$  is more than the radius of a neighbourhood, Lemma 5.1.1 state that no clique can span across  $A_j$  and  $A_k$  (contain site from both clusters). More importantly, this means that no clique can span across all three clusters  $A_i$ ,  $A_j$ , and  $A_k$ . Thus, the superedges  $\{\Delta_{A_i, A_l} : A_l \in \gamma A_i\}$ , contain *all* cliques that span across  $A_i$  and other nodes in  $\mathcal{A}'$ . Since this is true  $\forall A_i \in \mathcal{A}'$ , we have that  $\mathcal{C}_{S'} = \mathcal{C}_{G'}$ .  $\square$

This provides the justification for clustering as we did in Chapter 3. We can now also use Theorem 5.1.1 to formulate the following procedure for generating acyclic graphs.

1. Choose cutsets (in this case horizontal strips) to generate slices for encoding.
2. For each slice, cluster by sequentially grouping columns at least  $r_s$  in width from left to right. This ensures that the condition specified by Theorem 5.1.1 is satisfied. <sup>3</sup>

In Chapter 4, we mentioned that the minimum radius condition itself is not enough to ensure acyclic-ness due to the possibility of picking a “strange” cutset. Here, by inspection, since we are sequentially generating a linear line of clusters, there cannot be any loops or “doughnuts”, and hence the generated cluster graph must be acyclic.

## 5.2 Results and Discussion

The experimental results here are generated using cutsets of thickness  $r_s$  and slices of column spacing  $2r_s$ . Clusters are generated by grouping  $r_s$  columns within the slices.

---

<sup>3</sup>By exactly +1 in distance.



Each slice is treated as the cluster subgraph  $\mathcal{G}'$  that make up a cluster graph  $\mathcal{G}$  along with cutset  $L$ . Cutset  $L$  contains the sites in the strips immediately above and below the slice. Linear parameter potentials are assumed as well.

### 5.2.1 Some Results

The test image used is shown in Figure 5.1. We use a MRF with the diamond neighbourhood system shown in Figure 5.2. Estimates are made only for the parameters for the cliques listed in the Table 5.1. All other cliques types are assumed have zero parameter. The potentials on simple cliques are assumed to be of the “Ising” kind.

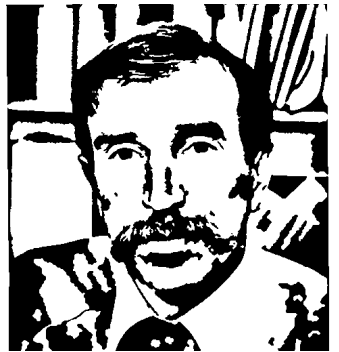


Figure 5.1: Test Image “opp2”

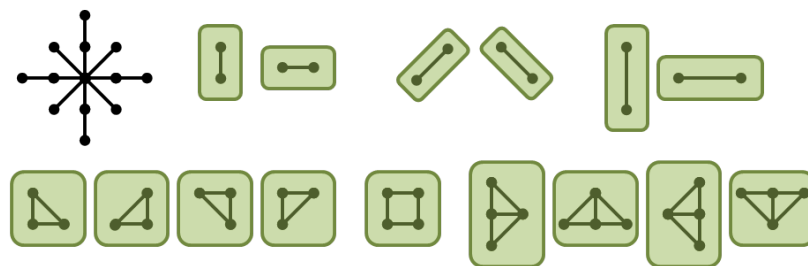


Figure 5.2: “Diamond” neighbourhood and some cliques.








Clique Type ( $i$ )	$\theta_i$
$i = 0$	
$i = 1$	
$i = 2$	
$i = 3$	
$i = 4$	
$i = 5$	
$i = 6$	

Table 5.1: Clique types used.

Using the parameters in Table 5.2 below, an overall coding rate of 0.0970 is achieved.

Clique Type ( $i$ )	$\theta_i$
$i = 0$	0.064
$i = 1$	0.062
$i = 2$	0.062
$i = 3$	0.063
$i = 4$	0.077
$i = 5$	0.072

Table 5.2: ML estimates using  $\theta_i = 0.5$  initial guess.

Using the parameters in Table 5.3 below, an overall coding rate of 0.1058 is achieved.

Clique Type ( $i$ )	$\theta_i$
$i = 0$	0.987
$i = 1$	1.109
$i = 2$	0.488
$i = 3$	0.316
$i = 4$	-0.132
$i = 5$	-0.068

Table 5.3: ML estimates using LS estimate as initial guess.

Both sets of parameters are ML estimates; they differ only in choice of initial guess.

Note that the cliques used are all of the simple type. We simply expanded the neighbourhood size, but have ignored the higher order cliques by setting their parameters to 0. We introduce the “square” clique, shown as clique type 6 in Table 5.1.

Using the parameters in Table 5.4 below, an overall coding rate of 0.1326 is achieved. A general “ranking” function is used for the type 6 clique potential to give higher probability to “homogeneous” configurations <sup>4</sup>.

---

<sup>4</sup>Referring to configurations with generally the same pixel phases across the clique. In the interest of time, we do not go over a detailed breakdown of our ranking function here.

Clique Type ( $i$ )	$\theta_i$
$i = 0$	1.610
$i = 1$	1.609
$i = 2$	-0.098
$i = 3$	-0.096
$i = 4$	0.028
$i = 5$	0.025
$i = 6$	-5.344

Table 5.4: ML estimates, including parameter for square clique. Square clique potential favours “homogeneous” configurations.

Using the parameters in Table 5.5 below, an overall coding rate of 0.0977 is achieved. The potential function of clique type 6 uses the same “ranking” approach mentioned above, but additionally, adds bias against “diagonal” patterns by reducing the rank for such configurations (Figure 5.3).

Clique Type ( $i$ )	$\theta_i$
$i = 0$	1.010
$i = 1$	0.995
$i = 2$	0.313
$i = 3$	0.308
$i = 4$	-0.097
$i = 5$	-0.348
$i = 6$	0.348

Table 5.5: ML estimates. Square clique potential contains specific bias against “diagonal” patterns.

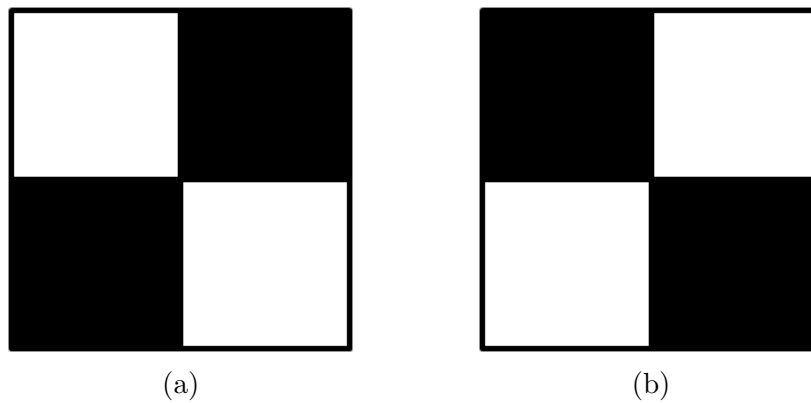


Figure 5.3: “Cross diagonal” patterns in a square clique.

Overall, when compared to the results from Figure 3.3 in Chapter 3, using the parameters in Table 5.5 does yield a slightly better coding rate for similar line spacing. We conclude that in this case, the image can be suitably encoded by MRFs with simple cliques (e.g. the Ising model), and that the introduction of complex cliques only yield a marginal improvement that does not justify its computation cost. Nonetheless, the

rate improvement shows that using complex cliques to capture larger structures can indeed be advantageous.

## 5.2.2 Discussion

### Local Maximums

By using 2 different starting “guesses”, MLE yield 2 completely different sets of parameters (Tables 5.2, and 5.3). This indicates that there are numerous local maxima on the likelihood. This is not surprising due to the increase in dimensionality. Also, by comparing the vertical and horizontal parameters (clique types 0,1,4,5) of Tables 5.2 with those of Table 5.3, we note that in the latter case exhibit significantly stronger interaction on “close” vertical/horizontal neighbours, but much weaker interaction on “far” ones. In some sense, we can interpret this as a case of spurious correlations caused by near/far neighbours of the same direction acting as facilitating/suppressing variables.

### Non-uniqueness of Potentials

In Chapter 2, it was briefly mentioned that the potentials are not unique. That is, different types of cliques, and different types of potential functions can, together, sum up to the same distribution over configurations; giving rise to the same MRF. Thus, deciding which “group” of potentials to use is not a trivial task, and determining which group suits best for a space of images is difficult as well.

## Computational Complexity

With more parameters to learn, the iterations needed for MLE with the gradient ascent method increase. This can be attributed in part to the increased number of statistics and computations needed for learning and Gibbs sampling. More importantly, as mentioned above, the increase in dimensionality results in a more difficult search on the likelihood. MLE's with strict stopping conditions may not have a solution, or may take multiple attempts and significantly more iterations to complete.

Also, increasing the neighbourhood system and using complex cliques result in larger clusters (if one adheres to a method which satisfies the minimum radius requirement). This means the super-alphabet for encoding the clusters grows large as well, increasing the computation time.

## 5.3 Future Work

### 5.3.1 Identifying Cliques and Potentials

With the introduction of complex cliques, one of the problems we observed is the non-uniqueness of potentials. This could be either potentials which vary in their parameters or entirely different potentials on entirely different cliques. For MLE, growing numbers of clique types results in a rapid increase in the difficulty of learning a desired set of parameters. This problem should be addressed on two fronts. First, as mentioned in Chapter 2, there has been past work regarding the uniqueness of potential distributions that arise [9, 10, 11]. Chapter 2 also mentioned that one way of doing this is to use *normalized potentials*, though this could be quite restricting to the types of potentials functions that can be used. Thus, one direction for future

work could be to identify standard ways by which potentials could be formulated to guarantee uniqueness. Second, to avoid the problem seen in the results of Table 5.2 and 5.3, we should try to use as few clique types as possible. Essentially, this amounts to identifying the minimal sufficient statistics (potentials) on the cliques which capture “most” features.

### 5.3.2 Coding and Computational Considerations

Due to the fact that the MLE computation time can be quite lengthy (especially for MRFs with complex cliques or high number of clique types), one practical consideration for coding purposes might be to use bins of MRFs. That is, learn the appropriate MRFs and parameters for some space of images offline, and classify the results into bins. The coding procedure could then be sped up by replacing the precoding phase with a supervised classification procedure that simply identifies the “bin” of parameters to use. This would also eliminate the cost of encoding MRF parameters.



# Bibliography

- [1] S. Amari. Information geometry on hierarchy of probability distributions. *IEEE Transactions on Information Theory*, 47(5):1701–1711, 2001.
- [2] S. Amari and H. Nagaoka. *Methods of Information Geometry*. Oxford University Press, 2007.
- [3] P. Brémaud. *Markov Chains, Gibbs Fields, Monte Carlo Simulation, and Queues*. Springer, 1999.
- [4] T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley & Sons, 2nd edition edition, 2006.
- [5] I. Csiszar.  $i$ -Divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 3(1):146–158, 1975.
- [6] J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5):1470–1480, 1972.
- [7] H. Derin and H. Elliott. Modeling and segmentation of noisy and textured images using Gibbs random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(1):39–55, 1987.

- [8] R.L. Dobrushin. Central limit theorem for non-stationary Markov chains i, ii. *Theory of Probability & Its Applications*, 1(1):65–80, 329–383, 1956.
- [9] R.L. Dobrushin. The problem of uniqueness of a Gibbsian random field and the problem of phase transitions. *Functional Analysis & Its Applications*, 2:302–312, 1968.
- [10] R.L. Dobrushin. Gibbsian random fields. The general case. *Functional Analysis & Its Applications*, 3:22–28, 1969.
- [11] R.L. Dobrushin. Prescribing a system of random variables by conditional distributions. *Theory of Probability & Its Applications*, 15(3):458–486, 1970.
- [12] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.
- [13] G.R. Grimmett. A theorem on random fields. *Bulletin of the London Mathematical society*, 5(1):81–84, 1973.
- [14] J.M. Hammersley and P. Clifford. Markov fields on finite graphs and lattices, 1971.
- [15] E. Ising. Beitrag zur theorie des ferromagnetismus. *Zeitschrift für Physik*, 31:253–258, 1925.
- [16] S. Kullback and R.A. Leibler. On information and sufficiency. *Ann. Math. Stat.*, 22(1):79–86, 1951.

- [17] B. McMillan. Two inequalities implied by unique decipherability. *Information Theory, IRE Transactions on*, 2(4):115–116, 1956.
- [18] University of Southern California. The USC-SIPI Image Database. <http://sipi.usc.edu/database/>.
- [19] F. Ono, W. Rucklidge, R. Arps, and C. Constantinescu. Jbig2 - The ultimate bi-level image coding standard. In *International Conference on Image Processing, 2000.*, volume 1, pages 140–143 vol.1, 2000.
- [20] R. Pasco. Source coding algorithms for fast data compression. Technical report, 1976.
- [21] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers, 1988.
- [22] R.E. Peierls. On Ising’s model of ferromagnetism. *Mathematical Proceedings of the Cambridge Philosophical Society*, 32:477–481, 1936.
- [23] M.G. Reyes. *Cutset Based Processing and Compression of Markov Random Fields*. PhD thesis, Electrical Engineering, University of Michigan, 2010.
- [24] J. Rissanen. Generalized Kraft inequality and arithmetic coding. *IBM J. Res. Develop*, 1976.
- [25] C.E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27:379–423, 623–, 1948.
- [26] G. Winkler. *Image Analysis, Random Fields, and Markov Chain Monte Carlo Methods: A Mathematical Introduction*. Springer, 2nd edition edition, 2003.