

Spectral-Spatial Analysis of
Remote Sensing Data:
An Image Model and
A Procedural Design

Varun Madhok
David Landgrebe

TR-ECE 99-10
August 1999

SCHOOL OF ELECTRICAL
AND COMPUTER ENGINEERING
PURDUE UNIVERSITY
WEST LAFAYETTE, INDIANA 47907-1285



TABLE OF CONTENTS

Page

ABSTRACT	v
1. INTRODUCTION	1
1.1 THESIS ORGANIZATION	2
PART ONE	5
2. LATTICE MODELS.....	7
2.1 PREVIOUS WORK.....	8
3. PROPOSED LATTICE MODEL	13
3.1 MODEL SOLUTIONS	16
3.2 TABULATION	28
4. MODEL IMPLEMENTATION.....	41
4.1 PREAMBLE.....	41
4.2 MODEL SIMULATION	44
4.2.1 Monte Carlo simulation.....	44
4.2.2 Construction of chain.....	47
4.2.3 Energy computation and the Metropolis algorithm.....	51
4.2.4 Random number generator (RNG)	52
4.2.5 Experimental results	53
4.3 APPLICATION TO MULTISPECTRAL ANALYSIS.....	55
4.3.1 Problem description and implementation.....	55
4.3.2 Algorithm for implementation.....	58
4.4 RATIONALE FOR USAGE – AN EXPERIMENTAL EVIDENCE	60
4.4.1 Experimental design	60
4.4.2 Results interpretation.....	65
4.5 EXPERIMENTS ON REMOTE SENSING DATA	75
4.5.1 DC flightline data	75
4.5.2 Forest data.....	80
PART TWO	83
5. INTERPRETING REMOTE SENSING DATA	85
5.1 PREVIOUS WORK.....	85
5.2 PRINCIPLES OF DATA FUSION.....	88
5.3 A PROCESS MODEL	94
6. A CASE STUDY - ANALYZING THE D.C. FLIGHTLINE.....	97
6.1 THE FUSION SUITE	98
6.1.1 Maximum likelihood classification.....	98
6.1.2 Unsupervised segmentation using the lattice model.....	99

6.1.3 Fusion - HYDICE data + Digital Elevation Map (DEM)	99
6.1.4 Decision tree (or graph) structure.....	99
6.1.5 Masking.....	100
6.1.6 Negative training	100
6.2 D.C. FLIGHTLINE ANALYSIS	100
6.2.1 Scrubbing - Removing bad data.....	100
6.2.2 Root Node - Statistical analysis	103
6.2.3 Node 1 - Separating WATER + SHADOW.....	106
6.2.4 Node 2 - Segmenting SHADOW	108
6.2.5 Node 3 - Separating GRASS and TREE	110
6.2.6 Node 4 - Extracting rooftops	113
6.2.7 Node 5 - Negative training on PATH.....	121
6.2.8 Nodes 6, 7 - Negative training on WATER2, WATER	124
6.2.9 Node 8 - Re-assigning SHADOW (spectral method).....	128
6.2.10 Node 9 - Negative training on classes WATER + WATER2.....	130
6.2.11 Node 9b - Reassigning class SHADOW (spatial scheme)	133
6.2.12 Synopsis	135
BIBLIOGRAPHY	139
APPENDICES	147
APPENDIX A – THE LATTICE MODELS.....	147
A.1 MATLAB code for solving 2×N model.	147
A.2 MATLAB code for solving 3×N model.....	147
A.3 MATLAB code for generating transfer matrices for L-ary 3×N model.	149
A.4 MATLAB code for computing magnetization M for 2×N model.....	149
A.5 MATLAB code for computing correlation C for 2×N model.....	150
A.6 MATLAB code for performance estimation for 2×N model.....	151
A.7 Correlation calculation for test images.....	152
A.8 Model characteristics and performance of estimators.....	152
GLOSSARY	159

ACKNOWLEDGMENTS

The author would like to thank Prof. David Landgrebe of the School of Electrical Engineering at Purdue University for his support of this work, and his guidance. Special thanks to Mr. Larry Biehl of the Multispectral Image Processing Laboratory at Purdue University for his patient assistance to the author in working in the Macintosh environment. Thanks are also due to Prof. Mark Bell, Prof. Charles Bouman, Prof. Carla Brodley and Prof. Antonio SaBarreto for the superior classroom instruction that laid the author's academic foundation.

Finally, thanks are due to Vijay, Asha and Siddharth Madhok for their support and their perseverance.

ABSTRACT

The distinguishing property of remotely sensed data is the multivariate information coupled with a two-dimensional pictorial representation amenable to visual interpretation. The contribution of this work is the design and implementation of various schemes that exploit this property.

This dissertation comprises two distinct parts. The essence of Part One is the algebraic solution for the partition function of a high-order lattice model of a two dimensional binary particle system. The contribution of Part Two is the development of a procedural framework to guide multispectral image analysis.

The characterization of binary (black and white) images with little semantic content is discussed in Part One. Measures of certain observable properties of binary images are proposed. A lattice model is introduced, the solution to which yields functional mappings from the model parameters to the measurements on the image. Simulation of the model is explained, as is its usage in the design of Bayesian priors to bias classification analysis of spectral data. The implication of such a bias is that spatially adjacent remote sensing data are identified as belonging to the same class with a high likelihood. Experiments illustrating the benefit of using the model in multispectral image analysis are also discussed.

The second part of this dissertation presents a procedural schema for remote sensing data analysis. It is believed that the data crucial to a successful analysis is provided by the human, as an interpretation of the image representation of the remote sensing spectral data. Subsequently, emphasis is laid on the design of an intelligent implementation of existing algorithms, rather than the development of new algorithms for analysis. The development introduces hyperspectral analysis as a problem requiring multi-source data fusion and presents a process model to guide the design of a solution. Part Two concludes with an illustration of the schema as used in the classification analysis of a given hyperspectral data set.

1. INTRODUCTION

This dissertation is a composite of various discussions on the usage of spatial information, available as image data, in the spectral analysis of remote sensing data.

Conventional remote sensing data analysis focuses on classifying the datum without incorporating information on the spatially adjacent data; i.e. the data is viewed not as an image but as an unordered listing of spectral measurements that can be shuffled arbitrarily without affecting analysis. There is a need to incorporate the image representation of the data in the analysis. Conversely, image processing in engineering literature is not directed to the analysis of multispectral data. The development of a nexus is one of the goals of this work.

Arguably, while a scientific solution to a given class of problems seeks robustness and broad applicability, practical concerns demand a certain level of customization to the solution for every case brought under consideration. Additionally, remote sensing data has the unique ability of being represented as an image. The subjective evaluations afforded by this can be used as an interface between the human and the computer - thus supplementing numeric data with analyst experience. These inferences are developed into a procedural framework, presented in the latter half of this dissertation, to initiate and guide the analysis of remote sensing data.

Part One introduces a model to explain the behavior of binary particle systems as a function of the labeling of the constituent elements. The equivalent of the particle system in multispectral image analysis is the 'hidden' classification of the spectral data, whose discovery is the goal of the analysis. Since the data has an image representation, the data-labeling is evident as a multipolar particle system arranged on a rectangular lattice. Part Two discusses a process model for the analysis of hyperspectral data. The foundation of the latter discussion is the proposition that all analysis is a fusion of diverse data/information, and the extraction of knowledge is dependent as much on the process model as it is on the design of the analysis algorithms.

There is a clean separation among the propositions of this work, and the respective presentations are divided between Part One (Chapters 2-4) and Part Two (Chapters 5-6).

1.1 Thesis Organization

Chapter 2 introduces the problem as the development of a mapping between the model parameters and the properties of a two-dimensional binary system. Chapter 3 proposes a lattice model as a solution to the problem. The model takes the form of a multi-chained loop with bipolar elements at each link. The system behavior depends on the interaction among the particles, as well as the orientations (± 1) of the particles. Three variants of the proposed model are discussed, and closed form solutions for the associated partition functions are found for each case. Based on these solutions, the mappings to the observable properties of the system are established. For practical implementation in software, these mappings are tabulated for use as look-up tables. Section 3.2 lists the relevant tables. Chapter 4 discusses a scheme for the implementation of the proposed lattice model(s) in the segmentation of spectral data. The link between the lattice modeling and multispectral image analysis is clarified in Section 4.1. One usage of the modeling is in the simulation of binary images with little or no semantic content. This aspect of the modeling is discussed in Section 4.2. The primary usage of the model is, however, in the design of biases for the statistical classification of spectral data. Section 4.3 discusses this usage. Section 4.4 presents an experiment to justify usage of the lattice model in classification analysis. The results of the experiment show that the error in classification is significantly reduced through the usage of the lattice model in the Bayesian analysis of the data. An explanation of this improvement is proposed in Section 4.4.2. Section 4.5 concludes Chapter 4 with experiments on real remote sensing data.

Chapter 5 discusses recent research on the design of procedural schemata for data fusion applications. It is concluded that the optimal engineering analysis uses the human to provide decision support for computer analysis. Chapter 5 also elaborates a process model developing on this claim. Some guidelines for the design of a successful data analysis are presented. Chapter 6 executes the strategies developed in the previous chapter through a detailed analysis of hyperspectral data collected for a flightline over Washington D.C.

The Appendices contains code for select software used in this work.

PART ONE

2. LATTICE MODELS

Statistical mechanics is the study of the statistical properties of systems comprising a large number of interacting elements. As opposed to the microscopic behavior of the system elements, the study focuses on the measurable macroscopic (visually perceptible) properties of the system. It is believed that, through certain assumptions on the system at the particle level, it is possible to algebraically model the macroscopic behavior of the system. The discovery of an algebraic mapping that adequately models a two-dimensional binary system is the goal of this work.

The systems of our interest are in the form of a two-dimensional lattice, and are binary in nature, i.e. the state space for the system elements is $\{-1, 1\}$. A visual representation of such a system would be a black and white image. Correspondingly, the macroscopic properties of the system are considered to be homogeneity¹ (or the nearest neighbor correlation), and the ratio of black to white pixels in the image. Chapter 3 presents a lattice model, and the function mapping the model parameters to the measurements of the system observables.

Applications of the proposed model to remote sensing data analysis are presented in Chapter 4. Remote sensing data analysis seeks to classify data based on the respective energy spectra. A popular solution of the problem is based on the assumption of the multivariate Gaussian distribution of the data [1] [2]. Such an analysis assesses each datum independent of the remainder of the data. However, spatially adjacent data elements have a strong likelihood of being functionally similar in spite of noise induced differences in the associated energy spectra. The failure to incorporate such prior knowledge in the analysis leads to data classification that is 'speckled'. Regularization theory [3] proposes that the solution of the problem should be smoothed via a bias in favor of more-likely configurations of the system. If the physics of the scene can be

simulated via an analytically tractable model, it becomes possible to bias the classification in favor of a spatially homogeneous (and realistic) configuration. Hence analysis can be directed via model 'priors'. The design of such priors is the primary goal of this section. It is believed that a single observation on the scene is sufficient to accurately estimate the intrinsic homogeneity and class distribution (black-white ratio) in the image. Where other models have regarded this as an intractable problem, we believe that the proposed model provides an exact and practical solution.

A discussion of other work in the area of system modeling is presented in Section 2.1. Note that the present discussion is in the area of low-level image modeling where we do not expect to see or to model macrostructure (content with a semantic description). Thus, a distinction may be made between time-series models [4] [5] [6] and random field models [7]. While adequate for modeling image texture, time-series models generally demonstrate directionality and periodicity, and are inappropriate for our application. A survey of various image models in different contexts is provided in [8]. The emphasis of this thesis is on Markov field models of images.

The actual model is presented in Chapter 3. The design and the implementation in regard to multispectral classification analysis are presented in sub-sections. Simulations demonstrating the efficacy of this implementation are presented in Section 3.3. Experiments with the model for the analysis of remote sensing data are described in Chapter 4.

2.1 Previous Work

A lattice model for interacting particle systems was first investigated by Ising as an explanation for ferromagnetism [9] [10]. The model proposed the magnetic crystal as consisting of a large number of points arranged on a long one-dimensional lattice (like a chain), with a positive or a negative spin associated with each point. In spite of various modifications and enhancements to the modeling, the general scheme has been propagated under Ising's name. Since the early days, the Ising model has been recognized

¹Visually, a black and white image with high homogeneity is perceived as containing large clusters of black and/or white pixels. Correspondingly, low homogeneity implies an image of granular appearance, with little clustering among pixels of any color.

as a stochastic process and, outside of statistical physics [9] [10] [11], is commonly known as a Markov Random Field (MRF) [12] [7]. In engineering applications, MRFs have gained popularity as a technique for image modeling [13] [14] [15] [16] [17] [18]. The proposed usage of the proposed lattice model is in a similar vein, albeit for remote sensing data analysis.

From a fundamental postulate in classical statistical mechanics, the probability distribution associated with the space of lattice configurations is directly proportional to the exponential of the 'energy' (in a manner of speaking) of that particular configuration [19], i.e. a Gibbs distribution. If X is a lattice configuration, or a given arrangement of 1's and -1's on the lattice, the probability of occurrence of this particular arrangement is given as

$$P_x(X) = \frac{1}{Z} e^{-E(X)}, \quad (2.1)$$

where Z is a normalizing constant known as the partition function, and $E(\bullet)$ is the 'energy' function operating on the space of all lattice configurations. A probability measure on the space of lattice configurations enables the image classification to be biased towards the configurations that have a high probability of occurrence. In pattern analysis literature various variants of this technique have been proposed. However, most of these applications have been deficient in a key respect. The energy of a lattice configuration is dependent on the value of the parameters of the energy function $E(\bullet)$. In most cases the parameter values are determined empirically [20] [21], or are approximated via a gradient search [16][17]. The presentation in Chapter 3 identifies the exact function-map between the model parameters and the measurable system properties; the measurements being obtained from a single observation of the system state.

Picard [22] discussed the problem of parameter inference for binary discrete Markov fields for equally likely classes. Although the solution is incomplete, the discussion in the article is representative of the issues associated with this problem. Geman and Geman [14] have demonstrated a computationally intensive method to calculate the model parameters via a stochastic gradient algorithm. However, the popular scheme for estimating distribution parameters is the method of maximum pseudo-likelihood [16] [17] [23] [24]. Another practical scheme has been proposed by Derin and

Elliott [25], and has been re-applied with modifications by Gurelli and Onural[26]. The latter scheme estimates the model parameters via the solution of a system of linear equations. Other solutions to the problem discard the pixel based approach in favor of that using ordered groups of pixels, cf. the window-based model [27], the line-process model of objects in the image [28], and the connected components model [29]. [30] have proposed an evolutionary approach to solving the model. Of interest also is the work of Tjelmeland and Besag [31], who design a Markov random field with higher order interactions.

As evident from the sampling above, the literature on Markov field application to image processing is quite extensive. Usually, parameter estimation is based on a search algorithm, and the solution is an approximation. This is largely a result of the lack of a closed form expression for the partition function (Z of Equation 2.1) for the two-dimensional Ising model. The breakthrough approximation for the partition function was the solution for the two dimensional Ising model provided by Onsager [32], who received the Nobel Prize in Physics for that work. Onsager's work was a great achievement in Physics in that it was the first explanation of phase transitions from order to disorder as a function of temperature. While critical to understanding ferromagnetism, the relevance of phase transitions to image modeling is unclear. The ability to model long-range order is also cited as a reason for favoring the two-dimensional Ising model over the one-dimensional Ising model. While the one-dimensional model can not model phase transitions, and does not exhibit long range order, it is much simpler to solve than higher dimensional models. The proposed model is a variant of the one-dimensional model. Experiments in Chapters 3 and 4 on real and simulated data show that this model performs adequately in the context of engineering image processing applications. It should also be mentioned that the desirable properties of the two-dimensional Ising model, are exhibited only asymptotically. It is unlikely that infinitely long lattices are appropriate models for remote sensing data analysis.

The proposed model takes the form of multiple chains with elemental interactions among neighbors within a chain, and with neighbors in the adjacent chain(s). A similar model was proposed by Qian and Titterington [33] and was described as a multidimensional Markov chain model. The said model performed adequately in

simulations of binary images. A survey of various models and their respective abilities to simulate images was presented in [34]. It should however be emphasized that a system model is not intended to exactly reproduce a system observation, but rather to replicate its properties. In the present discussion, the focus will be on replicating the macroscopic properties of various binary systems in their simulations.

While hyperspectral data has been quite adequate to the task of scene segmentation [2], it is believed that superior results can be obtained by incorporating spatial information on the data in the analysis [35]. [36] and [37] have proposed techniques that group data and grow regions based on spectral similarity. Markov fields have also been used in remote sensing analysis [38] [39] [40]. The implementation in this presentation is similar, the chief contribution being the use of the new model in the analysis. A limitation of the new model is its restriction to binary scenes, where data is labeled one of two classes. Direct application of this model is of limited value to real data. A solution to this issue is a hierarchical implementation of the solution. A coarse segmentation of the scene is used as input to the application, and the data with a given label is separated into two sub-classes. Such a scheme has alternately been proposed as decision tree type algorithms [41] [42] or multiple resolution segmentation [43] [44], in the analysis of image data.

3. PROPOSED LATTICE MODEL

The following analysis presents a model, or rather three variants of a central theme, for a binary particle system arranged on a lattice. The three variants differ in the design of particle interactions. A schematic presentation of each of the models is followed by an analysis of the respective properties. The analysis assumes a finite lattice structure in the form of a chain.

The notation used in this section is as below.

- N : Number of elements in lattice; equivalently, the number of links in the chain.
- L : Number of states possible for an element in the lattice. Unless specified otherwise, L is 2. Correspondingly, the state space for a lattice element will be denoted Λ ($=\{-1, 1\}$ unless specified otherwise).
- X : The system, an ordering of binary random variables. An observation of the system X is an arrangement of -1's and 1's on a lattice, and can be visualized as a black and white image. For ease of notation, X will represent both the stochastic process and the observation of the system state.
- q : The inter-particle attraction among nearest neighbors in the lattice. The magnitude of q determines the amount of coagulation or clustering seen in the image.
- h : The external field acting on the system. The sign and the magnitude of h determine the proportion of black (or white) content in the image.
- $E(\bullet)$: Given an observation of the system X , the energy of the configuration is measured as $E(X)$. $E(\bullet)$ is a function of q and h .
- Z : The partition function. Following the Hammersley-Clifford theorem [15] and Boltzmann [9], the probability distribution over the space of lattice configurations is Gibbsian. The density function is correspondingly given as

$$P_x(X) = \frac{1}{Z} e^{-E(X)}. \quad (3.1)$$

Z is the partition function, a normalizing factor, obtained as below.

$$Z = \sum_{\{X\}} e^{-E(X)}. \quad (3.2)$$

The summation in Equation 3.2 is over all possible realizations of the system. The issue of the energy function $E(\bullet)$ is critical to a model and the design is dependent on the system being modeled. The three models considered here, differ in the type of particle interaction, and thus in the choice of the energy function.

It can be shown that for a given lattice of length N , the partition function can be calculated as

$$\begin{aligned} Z &= \sum_{\{X\}} \exp(-E(X)) \\ &= \text{Tr}(\mathbf{A}^N) \end{aligned}$$

where \mathbf{A} is the transfer matrix representative of the system model, and $\text{Tr}(\bullet)$ is the trace function² on matrix operators. Details on the transfer matrix, and its relation to the partition function can be found in Thompson [Chapter 5, 10].

It follows that the partition function can be obtained using the eigenvalues of \mathbf{A} . If $\{l_i\}$ is the set of eigenvalues of \mathbf{A} then

$$\text{Tr}(\mathbf{A}^N) = \sum l_i^N.$$

Later development will show that \mathbf{A} is symmetric for the models considered. Consequently, the largest (principal) eigenvalue of \mathbf{A} is unique. For large N , a good estimate of the partition function is thus obtained by examining the principal eigenvalue.

By design, the energy function $E(\bullet)$ is a polynomial of first degree in q and h . It is easily shown (Equations A.1-2 - Appendix A.8, [45][22]) that (expected) pair-correlation, also called the internal energy per particle, C can be written as

$$C = \frac{1}{N} \sum_{\{X\}} \frac{dE(X)}{dq} P_x(X) = \frac{1}{N} \frac{d \log Z}{dq}. \quad (3.3)$$

Likewise, the (expected) magnetization per particle, M , can be computed as

$$M = \frac{1}{N} \sum_{\{X\}} \frac{dE(X)}{dh} P_x(X) = \frac{1}{N} \frac{d \log Z}{dh}. \quad (3.4)$$

The system model is thus the map

$$f : (q, h) \rightarrow (C, M). \quad (3.5)$$

² The trace of a matrix is the sum of its diagonal elements, or equivalently the sum of its eigenvalues.

Note that C and M are averages with respect to the distribution P_X . It follows that C and M can be estimated as sample averages from various observations on the given system. In the context of a black and white image, C is a measure of the granularity (large correlation translates to larger clumps of same-colored pixels), and M is a measure of the number of black pixels relative to that of the white pixels (or vice versa, depending on the notation). Given a tabulation of the function in Equation 3.5, these estimates can be used to infer q and h . Thus a parameterization at the pixel level (q, h) translates to macroscopic properties (C, M) of the system. The models presented here are for lattices of sizes $1 \times N$, $2 \times N$ and $3 \times N$ respectively (refer Figures 3.1-3). Note that the N th column is coupled to the first for each of the models. Each chain, μ , in the system X is composed of the elements μ_i for $i=1, \dots, N$. Each element μ_i is a binary (± 1) random variable, and interaction is restricted to nearest neighbors. The vertical bars represent interaction between elements in adjacent chains (if any).

The applicability of these lattice models to two dimensional image data systems will be examined in Chapter 4.

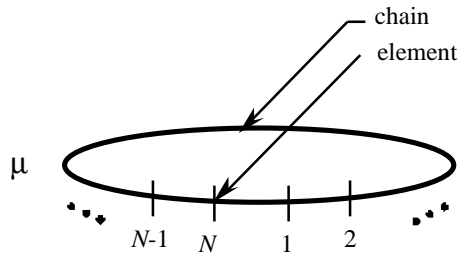


Fig. 3.1: The $1 \times N$ lattice model

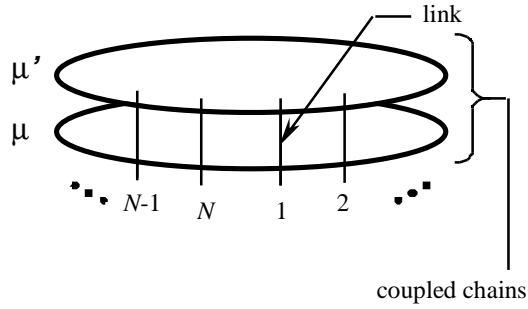


Fig. 3.2: The $2 \times N$ lattice model

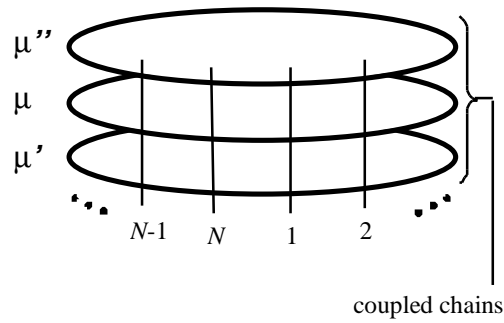


Fig. 3.3: The $3 \times N$ lattice model

3.1 Model Solutions

As stated earlier, the motivation of this work is the discovery of a closed form representation of the partition function Z , and thus, formulae for C and M in q and h .

In Figures 3.4-6, the first set of solutions is presented for the three models under the assumption that the external field is zero, i.e. $h=0$. This is equivalent to imposing equally likely states $(\{-1, 1\})$ on the bipolar elements. In Figure 3.7, this constraint on $h(=0)$ is removed, and a general solution is arrived at for the $2 \times N$ lattice model. Unfortunately, a similar analysis for the $3 \times N$ lattice requires the solution of fifth order polynomials and can not be carried through.

The procedure for the solution(s) is as below.

- The chain μ is defined. Additional chains, as per the model, are identified using one or more quotation marks ('). The system is represented in terms of one or more μ chains, as per the chosen model.
- The interaction among the elements is designed for each model, based on q (and h for Figure 3.7).
- The energy function $E(X)$ is formulated for each model.
- The corresponding transfer matrix \mathbf{A} is found. (Recall that $Z = \text{Tr}(\mathbf{A}^N)$ [10]).
- The characteristic polynomial for each transfer matrix is calculated, and factorized using Maple [46]. In cases where Maple returned complex roots of the characteristic polynomial, the method of Tartaglia [47] is used to manually compute the solutions for cubic polynomials. These solutions are the eigenvalues $\{l_i\}$ of the transfer matrix \mathbf{A} .
- Z is approximated in terms of the principal eigenvalue of \mathbf{A} [10].
- The correlation C is calculated using Equation 3.3.
- The magnetization M is calculated using Equation 3.4.

Note: In Figures 3.4-7, subscripts are used with the notation for partition function, correlation and magnetization, to identify the expression with the respective model. For instance, $Z_{1,N}$ denotes the partition function for the $1 \times N$ model.

$$\begin{aligned}
 Z_{1,N} &= \text{Tr}(\mathbf{A}^N) \\
 &= \sum_{i=1}^L l_i^N \\
 &= \exp(-Nq) \left[(\exp(2q) + L - 1)^N + (L - 1)(\exp(2q) - 1)^N \right]. \\
 C_{1,N} &= \frac{1}{NZ_{1,N}} \frac{dZ_{1,N}}{dq} \\
 &= -1 + \frac{2 \exp(2q)}{\exp(2q) + L - 1} \left[\frac{1 + (L - 1) \left(\frac{\exp(2q) - 1}{\exp(2q) + L - 1} \right)^{N-1}}{1 + (L - 1) \left(\frac{\exp(2q) - 1}{\exp(2q) + L - 1} \right)^N} \right] \\
 &\approx -1 + \frac{2 \exp(2q)}{\exp(2q) + L - 1}.
 \end{aligned}$$

Fig. 3.4 (contd.): Solution of the $1 \times N$ lattice model for external field $h=0$ – partition function $Z_{1,N}$, correlation $C_{1,N}$.

$$\begin{aligned}
 l_1 &= \frac{a^4 - 1}{a^3}, \\
 l_2 &= \frac{a^4 - 1}{a}, \\
 l_3 &= \frac{a^2 + 1}{2a^3} \left[a^4 + 1 + (a^8 + 10a^4 + 1 - 4a^6 - 4a^2)^{0.5} \right], \\
 l_4 &= \frac{a^2 + 1}{2a^3} \left[a^4 + 1 - (a^8 + 10a^4 + 1 - 4a^6 - 4a^2)^{0.5} \right]. \\
 Z_{2,N} &= \text{Tr}(\mathbf{A}^N) \\
 &= \sum_{j=1}^4 l_j^N \approx_3 l_3^N. \\
 b &= (a^2 + 10a^{-2} + a^{-6} - 4 - 4a^{-4})^{0.5}, \\
 \frac{db}{da} &= \frac{1}{2b} (2a - 20a^{-3} - 6a^{-7} + 16a^{-5}), \\
 \frac{dl_3}{da} &= a(a + a^{-3} + b) + \frac{a^2 + 1}{2} \left(1 - 3a^{-4} + \frac{db}{da} \right), \\
 C_{2,N} &= \frac{1}{NZ_{2,N}} \frac{dZ_{2,N}}{dq} \approx \frac{a}{l_3} \frac{dl_3}{da}.
 \end{aligned}$$

Fig. 3.5 (contd.): Solution of the $2 \times N$ lattice model for external field $h=0$ – principal eigenvalue l_3 , partition function $Z_{2,N}$, and correlation $C_{2,N}$.

$$\begin{aligned}
 a &= e^q, \\
 m_1 &= \frac{a^4 - 1}{3a^5}, \\
 m_2 &= \frac{a^2 + 1}{3a^5}, \\
 p_1 &= -12a^{16} + 12a^{14} + 80a^{12} - 120a^{10} - 120a^8 + 80a^6 + 12a^4 - 12a^2 \\
 &\quad + 8 + 8a^{18}, \\
 q_1 &= 12a^2 \left(\begin{aligned} &3 - 18a^2 + 51a^4 + 12a^8 - 60a^6 + 126a^{12} + 84a^{10} + 126a^{16} \\ &+ 12a^{20} - 396a^{14} + 51a^{24} + 3a^{28} - 18a^{26} - 60a^{22} + 84a^{18} \end{aligned} \right)^{0.5}, \\
 p_2 &= \left(\begin{aligned} &8 - 36a^{22} + 72a^{18} - 36a^{14} - 36a^{10} + 72a^6 - 36a^2 - 96a^{16} \\ &+ 592a^{12} - 96a^8 + 8a^{24} + 48a^{20} + 48a^4 \end{aligned} \right), \\
 q_2 &= 12a^2 \left(\begin{aligned} &3a^{40} - 24a^2 + 3 + 84a^4 + 6a^8 - 132a^6 - 306a^{12} + 336a^{10} + 1287a^{16} \\ &- 2148a^{20} - 744a^{14} + 1287a^{24} - 306a^{28} + 84a^{36} + 6a^{32} + 336a^{30} \\ &- 132a^{34} - 24a^{38} - 744a^{26} + 564a^{22} + 564a^{18} \end{aligned} \right)^{0.5}.
 \end{aligned}$$

For $j=1,2$

$$\begin{aligned}
 \theta_j &= \frac{1}{3} \tan^{-1} \frac{q_j}{p_j} + \cos^{-1} \text{sign}(p_j), \\
 x_j &= (p_j^2 + q_j^2)^{1/6} \cos \theta_j, \\
 y_j &= (p_j^2 + q_j^2)^{1/6} \sin \theta_j. \\
 l_1 &= \frac{1 - a^2 - a^4 + a^6}{a^3}, \\
 l_2 &= \frac{1 - a^2 + a^4 + a^6}{a^3}, \\
 l_3 &= m_1(x_1 + 1 + a^2 + a^4 + a^6), \\
 l_7 &= m_1 \left(-\frac{x_1}{2} + 1 + a^2 + a^4 + a^6 + \frac{\sqrt{3}y_1}{2} \right), \\
 l_8 &= m_1 \left(-\frac{x_1}{2} + 1 + a^2 + a^4 + a^6 - \frac{\sqrt{3}y_1}{2} \right).
 \end{aligned}$$

Fig. 3.6 (contd.): Solution of the $3 \times N$ lattice model for external field $h=0$.

$$\begin{aligned}
 l_4 &= m_2(x_2 + 1 + 2a^4 + a^8), \\
 l_5 &= m_2\left(-\frac{x_2}{2} + 1 + 2a^4 + a^8 + \frac{\sqrt{3}y_2}{2}\right), \\
 l_6 &= m_2\left(-\frac{x_2}{2} + 1 + 2a^4 + a^8 - \frac{\sqrt{3}y_2}{2}\right). \\
 Z_{3,N} &= \sum_{j=1}^8 l_j^N \approx l_4^N. \\
 \frac{dm_2}{da} &= \frac{1}{3}(-3a^{-4} - 5a^{-6}). \\
 \frac{dp_2}{da} &= \left(\begin{array}{l} -792a^{21} + 1296a^{17} - 504a^{13} - 360a^9 + 432a^5 - 72a - 1536a^{15} + 7104a^{11} \\ -768a^7 + 192a^{23} + 960a^{19} + 192a^3 \end{array} \right) \\
 \frac{dq_2}{da} &= \frac{2q_2}{a} + \left(\frac{3}{q_2}\right)^{0.5} \left(\begin{array}{l} 120a^{40} - 48a^2 + 336a^4 + 48a^8 - 792a^6 - 3672a^{12} + 3360a^{10} \\ +20592a^{16} - 42960a^{20} - 10416a^{14} + 30888a^{24} - 8568a^{28} \\ +3024a^{36} + 192a^{32} + 10080a^{30} - 4488a^{34} - 912a^{38} \\ -19344a^{26} + 12408a^{22} + 10152a^{18} \end{array} \right) \\
 \frac{dx_2}{da} &= \frac{1}{6}(p_2^2 + q_2^2)^{-5/6} \left[2p_2 \frac{dp_2}{da} + 2q_2 \frac{dq_2}{da} \right] \text{sign}(p_2) |\cos\theta_2| \\
 &\quad - \frac{1}{3}(p_2^2 + q_2^2)^{1/6} \sin\theta_2 \frac{p_2 \frac{dq_2}{da} - q_2 \frac{dp_2}{da}}{p_2^2 + q_2^2} \\
 \frac{dl_4}{da} &= \frac{dm_2}{da} (x_2 + 1 + 2a^4 + a^8) + m_2 \left(\frac{dx_2}{da} + 8a^3 + 8a^7 \right) \\
 C_{3,N} &= \frac{1}{NZ_{3,N}} \frac{dZ_{3,N}}{dq} \approx \frac{a}{l_4} \frac{dl_4}{da}.
 \end{aligned}$$

Fig. 3.6 (contd.): Solution of the $3 \times N$ lattice model for external field $h=0$ – principal eigenvalue l_4 , partition function $Z_{3,N}$, and correlation $C_{3,N}$.

$$\begin{aligned}
 \dot{l}_2 &= \frac{A}{3t} - t, \\
 \dot{l}_3 &= -\frac{\dot{l}_2}{2} + \frac{1}{2}(-4A - 3\dot{l}_2^2)^{0.5}, \\
 \dot{l}_4 &= -\frac{\dot{l}_2}{2} - \frac{1}{2}(-4A - 3\dot{l}_2^2)^{0.5}. \\
 l_1 &= e^q - e^{-3q}, \\
 l_2 &= \frac{\bar{l}_2 - x_2}{3}, \\
 l_3 &= \frac{\bar{l}_3 - x_2}{3}, \\
 l_4 &= \frac{\bar{l}_4 - x_2}{3}. \\
 \frac{dl_3}{ds} &= \frac{1}{3} \frac{d\bar{l}_3}{ds} - \frac{1}{3} \frac{dx_2}{ds}, \\
 \frac{d\bar{l}_3}{ds} &= -\frac{1}{2} \frac{d\bar{l}_2}{ds} + \frac{1}{4}(-4A - 3\dot{l}_2^2)^{-0.5} \left(-4 \frac{dA}{ds} - 6\dot{l}_2 \frac{d\bar{l}_2}{ds} \right), \\
 \frac{d\bar{l}_2}{ds} &= \frac{1}{3t} \frac{dA}{ds} - \frac{dt}{ds} \left(\frac{A}{3t^2} + 1 \right), \\
 \frac{dt}{ds} &= \frac{1}{54t^2} \left[-9 \frac{dB}{ds} + \frac{81B \frac{dB}{ds} + 18A^2 \frac{dA}{ds}}{(81B^2 + 12A^3)^{0.5}} \right]. \\
 \frac{dA}{dh} &= 12(e^{4q} - 5) \sinh 2h - 24e^{6q} \sinh 4h, \\
 \frac{dA}{dq} &= 24e^{4q} \cosh 2h + 36 \cosh 6h - 12 \sinh 2h - 36e^{6q} \cosh 4h + 24e^{-2q}. \\
 \frac{dx_2}{dh} &= -4e^{3q} \sinh 2h, \\
 \frac{dx_2}{dq} &= -6e^{3q} \cosh 2h - e^q + 3e^{-3q}.
 \end{aligned}$$

Fig. 3.7 (contd.): General solution of the $2 \times N$ lattice model – principal eigenvalue l_3 .

$$\frac{dB}{dh} = - \left[\begin{array}{l} 36(e^{9q} + e^{5q} - e^q - e^{-3q})\sinh 2h + 72(e^{7q} - e^{3q})\sinh 4h \\ + 6x_2^2 \frac{dx_2}{dh} \end{array} \right]$$

$$\frac{dB}{dq} = - \left[\begin{array}{l} 18(9e^{9q} + 5e^{5q} - e^q + 3e^{-3q})\cosh 2h + 18(7e^{7q} - 3e^{3q})\cosh 4h \\ - 90e^{-5q} + 216e^{3q} + 90e^{-q} + 6x_2^2 \frac{dx_2}{dq} \end{array} \right]$$

$$Z_{2,N} = \text{Tr}(\mathbf{A}^N)$$

$$= \sum_{i=1}^4 l_i^N \approx l_3^N$$

$$C_{2,N} = \frac{1}{NZ_{2,N}} \frac{dZ_{2,N}}{dq} \approx \frac{1}{l_3} \frac{dl_3}{dq}$$

$$M_{2,N} = \frac{1}{NZ_{2,N}} \frac{dZ_{2,N}}{dh} \approx \frac{1}{l_3} \frac{dl_3}{dh}$$

Fig. 3.7 (contd.): General solution of the $2 \times N$ lattice model – partition function $Z_{2,N}$, correlation $C_{2,N}$, and magnetization $M_{2,N}$.

3.2 Tabulation

Although convenient, the expressions derived in Figures 3.4-7 can not be used directly in any practical application.

Table 3.1 lists the q - C map for the $2 \times N$ model for $h=0$, as obtained in Figure 3.5. Tables 3.2-3 respectively list the C and M over a range of (q, h) , as obtained via the formulae in Figure 3.7. While C and M can not be calculated exactly, they can be estimated as sample averages over a set of observations ($\{V\}$) over the lattice system. The set of observations is composed of several different realizations of chains over the image lattice. For a given observation on a binary $2 \times N$ lattice chain, using notation consistent with the previous section, C and M are be estimated as below. Note the use of normalizing constants to restrict the estimates to $[0, 1]$.

$$\hat{C} = \frac{1}{|V|} \sum_{\{V\}} \left[\frac{1}{3N} \sum_{i=1}^N (\mu_i \mu_{i+1} + \mu_i' \mu_{i+1}' + \mu_i \mu_i') \right]$$

$$\hat{M} = \frac{1}{|V|} \sum_{\{V\}} \left[\frac{1}{2N} \sum_{i=1}^N (\mu_i + \mu_i') \right]$$

The accuracy of these estimators is discussed in Appendix A.8.

Given the estimates of magnetization and correlation for a given system, the tables can be used to look up (estimates of) the corresponding q and h .

Table 3.1
Average correlation per pixel, $C_{2,N}$ (for $h=0$) $2 \times N$ model.

q	$C_{2,N}$
0.0	0.000
0.1	0.101
0.2	0.208
0.3	0.325
0.4	0.453
0.5	0.586
0.6	0.708
0.7	0.807
0.8	0.878
0.9	0.925
1.0	0.954
1.1	0.972
1.2	0.983
1.3	0.989
1.4	0.993
1.5	0.996
1.6	0.997
1.7	0.998
1.8	0.999
1.9	0.9995
2.0	1.0

Table 3.2
Average correlation per pixel, $C_{2,N}$, for general $2 \times N$ model

$q \downarrow h \Rightarrow$	0.0001	0.0500	0.1000	0.1500	0.2000	0.2500	0.3000	0.3500
0.0001	0.0001	0.0026	0.0101	0.0223	0.0391	0.0602	0.0850	0.1133
0.0500	0.0502	0.0533	0.0624	0.0774	0.0977	0.1228	0.1521	0.1849
0.1000	0.1011	0.1049	0.1163	0.1346	0.1593	0.1893	0.2236	0.2612
0.1500	0.1534	0.1583	0.1725	0.1952	0.2251	0.2606	0.3003	0.3425
0.2000	0.2079	0.2141	0.2320	0.2601	0.2961	0.3376	0.3822	0.4280
0.2500	0.2650	0.2730	0.2956	0.3302	0.3728	0.4197	0.4680	0.5155
0.3000	0.3251	0.3354	0.3640	0.4057	0.4546	0.5055	0.5550	0.6015
0.3500	0.3880	0.4013	0.4371	0.4861	0.5395	0.5915	0.6393	0.6818
0.4000	0.4533	0.4705	0.5142	0.5691	0.6239	0.6735	0.7163	0.7529
0.4500	0.5198	0.5418	0.5932	0.6510	0.7032	0.7470	0.7829	0.8125
0.5000	0.5859	0.6134	0.6708	0.7272	0.7731	0.8091	0.8375	0.8603
0.5500	0.6494	0.6830	0.7431	0.7936	0.8310	0.8588	0.8802	0.8972
0.6000	0.7085	0.7481	0.8062	0.8479	0.8765	0.8971	0.9127	0.9250
0.6500	0.7616	0.8062	0.8580	0.8901	0.9109	0.9256	0.9368	0.9455
0.7000	0.8077	0.8554	0.8981	0.9215	0.9362	0.9465	0.9543	0.9605
0.7500	0.8466	0.8950	0.9278	0.9443	0.9544	0.9616	0.9671	0.9714
0.8000	0.8786	0.9253	0.9493	0.9605	0.9675	0.9724	0.9762	0.9793
0.8500	0.9045	0.9477	0.9644	0.9720	0.9768	0.9802	0.9828	0.9850
0.9000	0.9251	0.9636	0.9750	0.9801	0.9834	0.9857	0.9876	0.9891
0.9500	0.9414	0.9747	0.9824	0.9858	0.9880	0.9897	0.9910	0.9921
1.0000	0.9541	0.9825	0.9876	0.9899	0.9914	0.9925	0.9935	0.9942
1.0500	0.9641	0.9878	0.9912	0.9927	0.9938	0.9946	0.9952	0.9958
1.1000	0.9718	0.9915	0.9937	0.9948	0.9955	0.9961	0.9965	0.9969
1.1500	0.9779	0.9940	0.9955	0.9962	0.9967	0.9971	0.9975	0.9978
1.2000	0.9826	0.9958	0.9968	0.9973	0.9976	0.9979	0.9982	0.9984
1.2500	0.9863	0.9970	0.9977	0.9980	0.9983	0.9985	0.9986	0.9988
1.3000	0.9891	0.9979	0.9983	0.9986	0.9987	0.9989	0.9990	0.9991
1.3500	0.9914	0.9985	0.9988	0.9990	0.9991	0.9992	0.9993	0.9993
1.4000	0.9932	0.9989	0.9991	0.9992	0.9993	0.9994	0.9995	0.9995
1.4500	0.9946	0.9992	0.9994	0.9994	0.9995	0.9996	0.9996	0.9996

Table 3.2 (contd.)
Average correlation per pixel, $C_{2,N}$, for general $2 \times N$ model

$q \downarrow h \Rightarrow$	0.4000	0.4500	0.5000	0.5500	0.6000	0.6500	0.7000	0.7500
0.0001	0.1446	0.1782	0.2138	0.2508	0.2887	0.3271	0.3655	0.4037
0.0500	0.2205	0.2581	0.2971	0.3368	0.3767	0.4162	0.4551	0.4928
0.1000	0.3010	0.3422	0.3838	0.4251	0.4656	0.5049	0.5425	0.5784
0.1500	0.3859	0.4294	0.4721	0.5134	0.5528	0.5901	0.6250	0.6576
0.2000	0.4734	0.5174	0.5593	0.5985	0.6350	0.6687	0.6997	0.7279
0.2500	0.5607	0.6029	0.6417	0.6772	0.7093	0.7383	0.7645	0.7880
0.3000	0.6438	0.6820	0.7162	0.7465	0.7735	0.7974	0.8187	0.8375
0.3500	0.7192	0.7518	0.7802	0.8050	0.8267	0.8457	0.8624	0.8771
0.4000	0.7839	0.8104	0.8330	0.8525	0.8693	0.8840	0.8967	0.9079
0.4500	0.8370	0.8576	0.8750	0.8898	0.9025	0.9136	0.9232	0.9315
0.5000	0.8789	0.8943	0.9073	0.9184	0.9278	0.9360	0.9431	0.9493
0.5500	0.9109	0.9223	0.9318	0.9399	0.9469	0.9529	0.9581	0.9626
0.6000	0.9349	0.9431	0.9500	0.9559	0.9610	0.9653	0.9692	0.9725
0.6500	0.9526	0.9585	0.9635	0.9677	0.9714	0.9746	0.9773	0.9798
0.7000	0.9656	0.9698	0.9734	0.9764	0.9791	0.9814	0.9834	0.9851
0.7500	0.9750	0.9780	0.9806	0.9828	0.9847	0.9863	0.9878	0.9891
0.8000	0.9818	0.9840	0.9858	0.9874	0.9888	0.9900	0.9910	0.9920
0.8500	0.9868	0.9883	0.9896	0.9908	0.9918	0.9926	0.9934	0.9941
0.9000	0.9904	0.9915	0.9924	0.9932	0.9940	0.9946	0.9952	0.9957
0.9500	0.9930	0.9938	0.9945	0.9950	0.9956	0.9960	0.9964	0.9968
1.0000	0.9949	0.9954	0.9959	0.9964	0.9967	0.9971	0.9974	0.9976
1.0500	0.9963	0.9967	0.9970	0.9973	0.9976	0.9979	0.9981	0.9983
1.1000	0.9973	0.9976	0.9978	0.9980	0.9982	0.9984	0.9986	0.9987
1.1500	0.9980	0.9982	0.9984	0.9986	0.9987	0.9988	0.9990	0.9991
1.2000	0.9985	0.9987	0.9988	0.9989	0.9990	0.9991	0.9992	0.9993
1.2500	0.9989	0.9990	0.9991	0.9992	0.9993	0.9994	0.9994	0.9995
1.3000	0.9992	0.9993	0.9994	0.9994	0.9995	0.9995	0.9996	0.9996
1.3500	0.9994	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997
1.4000	0.9996	0.9996	0.9997	0.9997	0.9997	0.9997	0.9998	0.9998
1.4500	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998

Table 3.3
Average magnetization per pixel, $M_{2,N}$, for general $2 \times N$ model

$q \downarrow h \Rightarrow$	0.0001	0.0500	0.1000	0.1500	0.2000	0.2500	0.3000	0.3500
0.0001	0.0001	0.0501	0.0998	0.1490	0.1975	0.2451	0.2915	0.3366
0.0500	0.0001	0.0584	0.1162	0.1731	0.2288	0.2828	0.3348	0.3846
0.1000	0.0001	0.0687	0.1364	0.2025	0.2664	0.3274	0.3853	0.4395
0.1500	0.0002	0.0815	0.1614	0.2385	0.3116	0.3801	0.4433	0.5011
0.2000	0.0002	0.0978	0.1928	0.2826	0.3657	0.4411	0.5084	0.5679
0.2500	0.0002	0.1187	0.2320	0.3363	0.4291	0.5098	0.5789	0.6375
0.3000	0.0003	0.1454	0.2809	0.4004	0.5013	0.5842	0.6516	0.7061
0.3500	0.0004	0.1797	0.3411	0.4746	0.5796	0.6602	0.7220	0.7696
0.4000	0.0005	0.2235	0.4130	0.5566	0.6596	0.7329	0.7858	0.8249
0.4500	0.0006	0.2787	0.4955	0.6413	0.7356	0.7976	0.8400	0.8703
0.5000	0.0007	0.3464	0.5844	0.7225	0.8023	0.8512	0.8833	0.9057
0.5500	0.0010	0.4263	0.6731	0.7939	0.8567	0.8931	0.9164	0.9323
0.6000	0.0012	0.5157	0.7544	0.8521	0.8985	0.9244	0.9407	0.9518
0.6500	0.0016	0.6090	0.8228	0.8964	0.9292	0.9470	0.9581	0.9658
0.7000	0.0020	0.6988	0.8760	0.9286	0.9510	0.9630	0.9705	0.9757
0.7500	0.0026	0.7781	0.9151	0.9512	0.9662	0.9742	0.9792	0.9828
0.8000	0.0033	0.8426	0.9426	0.9668	0.9766	0.9819	0.9853	0.9877
0.8500	0.0042	0.8915	0.9614	0.9773	0.9838	0.9873	0.9896	0.9913
0.9000	0.0053	0.9267	0.9741	0.9845	0.9887	0.9911	0.9926	0.9938
0.9500	0.0066	0.9510	0.9825	0.9893	0.9921	0.9937	0.9947	0.9955
1.0000	0.0083	0.9675	0.9882	0.9926	0.9945	0.9955	0.9962	0.9968
1.0500	0.0104	0.9784	0.9920	0.9949	0.9961	0.9968	0.9973	0.9977
1.1000	0.0130	0.9857	0.9945	0.9964	0.9972	0.9977	0.9981	0.9983
1.1500	0.0163	0.9905	0.9962	0.9975	0.9980	0.9984	0.9986	0.9988
1.2000	0.0203	0.9937	0.9974	0.9982	0.9986	0.9988	0.9990	0.9991
1.2500	0.0252	0.9957	0.9982	0.9987	0.9990	0.9992	0.9993	0.9994
1.3000	0.0312	0.9971	0.9987	0.9991	0.9993	0.9994	0.9995	0.9995
1.3500	0.0387	0.9981	0.9991	0.9994	0.9995	0.9996	0.9996	0.9997
1.4000	0.0479	0.9987	0.9994	0.9995	0.9996	0.9997	0.9997	0.9997
1.4500	0.0591	0.9991	0.9996	0.9997	0.9997	0.9998	0.9998	0.9998

Table 3.3 (contd.)
Average magnetization per pixel, $M_{2,N}$, for general $2 \times N$ model

$q \downarrow h \Rightarrow$	0.8000	0.8500	0.9000	0.9500	1.0000	1.0500	1.1000	1.1500
0.0001	0.6642	0.6912	0.7165	0.7399	0.7617	0.7819	0.8006	0.8179
0.0500	0.7190	0.7440	0.7670	0.7880	0.8073	0.8249	0.8410	0.8557
0.1000	0.7705	0.7927	0.8128	0.8310	0.8474	0.8622	0.8755	0.8876
0.1500	0.8170	0.8360	0.8528	0.8679	0.8814	0.8934	0.9041	0.9137
0.2000	0.8572	0.8727	0.8864	0.8985	0.9092	0.9187	0.9271	0.9346
0.2500	0.8906	0.9029	0.9136	0.9231	0.9314	0.9387	0.9451	0.9508
0.3000	0.9175	0.9269	0.9351	0.9423	0.9486	0.9541	0.9590	0.9633
0.3500	0.9384	0.9455	0.9517	0.9571	0.9618	0.9659	0.9696	0.9728
0.4000	0.9544	0.9597	0.9643	0.9682	0.9717	0.9748	0.9775	0.9798
0.4500	0.9664	0.9703	0.9737	0.9766	0.9791	0.9814	0.9834	0.9851
0.5000	0.9754	0.9782	0.9807	0.9828	0.9847	0.9863	0.9877	0.9890
0.5500	0.9820	0.9840	0.9858	0.9874	0.9887	0.9899	0.9910	0.9919
0.6000	0.9868	0.9883	0.9896	0.9907	0.9917	0.9926	0.9934	0.9941
0.6500	0.9904	0.9914	0.9924	0.9932	0.9939	0.9946	0.9951	0.9956
0.7000	0.9930	0.9937	0.9944	0.9950	0.9955	0.9960	0.9964	0.9968
0.7500	0.9949	0.9954	0.9959	0.9963	0.9967	0.9971	0.9974	0.9976
0.8000	0.9962	0.9966	0.9970	0.9973	0.9976	0.9978	0.9981	0.9983
0.8500	0.9972	0.9975	0.9978	0.9980	0.9982	0.9984	0.9986	0.9987
0.9000	0.9980	0.9982	0.9984	0.9985	0.9987	0.9988	0.9989	0.9990
0.9500	0.9985	0.9987	0.9988	0.9989	0.9990	0.9991	0.9992	0.9993
1.0000	0.9989	0.9990	0.9991	0.9992	0.9993	0.9994	0.9994	0.9995
1.0500	0.9992	0.9993	0.9994	0.9994	0.9995	0.9995	0.9996	0.9996
1.1000	0.9994	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997
1.1500	0.9996	0.9996	0.9996	0.9997	0.9997	0.9997	0.9998	0.9998
1.2000	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998
1.2500	0.9998	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999
1.3000	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.3500	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.4000	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	1.0000
1.4500	0.9999	0.9999	0.9999	0.9999	1.0000	1.0000	1.0000	1.0000

Though simplistic, the above discussion indicates that remote sensing data is not an unordered collection of spectral measurements, and that spatial information can be a useful adjunct to the spectral data. This chapter will present a scheme to use the labeling of the analyzed element's neighborhood to bias the multispectral classification of the said element. The identification of the magnitude and the direction of the applied bias is thus the critical issue. Consider Figure 4.2. Starting at the top left and going clockwise around the figure, the four peripheral images may be characterized as below:

Mainly 'white', with isolated instances of 'black'.

Mainly 'black', with isolated instances of 'white'.

Even distribution of 'black' and 'white' in large clusters of pixels.

Even distribution of 'black' and 'white' pixels with little clustering.

In each case, the element to be labeled is shown in gray and has two of its neighbors pre-labeled as black. (For clarity in presentation, this example focuses only on the immediate neighbors to the North and to the West of the said pixel). The image in the center is an isolated representation of the pixels relevant to the analysis.

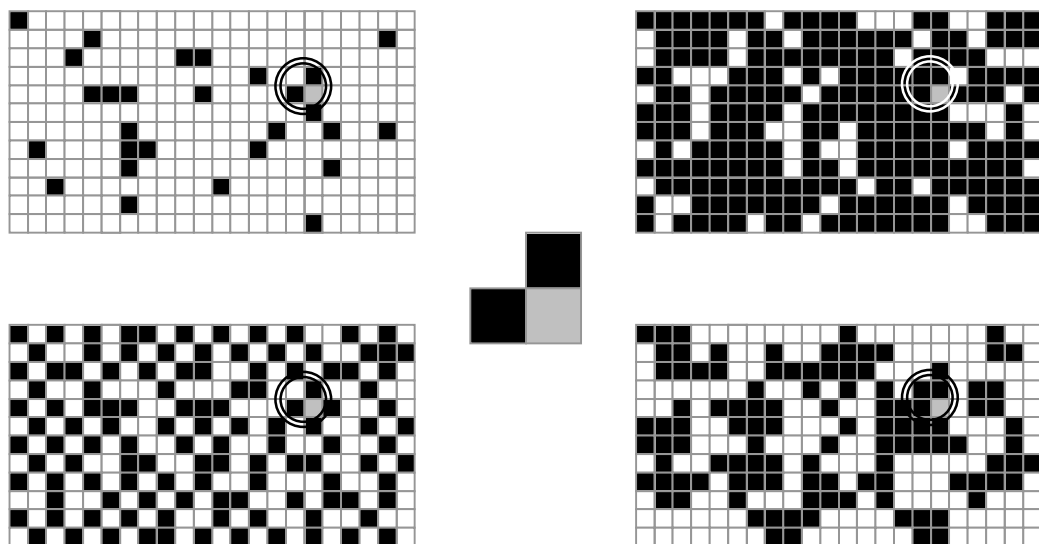


Fig. 4.2: Central image illustrates a case where the pixel under analysis has two neighbors pre-labeled as 'black'. The four peripheral images present scenarios that could possibly yield the situation depicted in the center.

The classification of the gray pixel in each image is sought via the relevant multispectral data and the labeling of its two identified neighbors. As reasoned previously, the observation of 'black' neighbors to the examined pixel should result in a biasing of the decision rule to favor a 'black' classification. However, from Figure 4.2 it may be rationalized that the bias is small for the images at the top left and at the bottom right. This reasoning is based on the fact that there appears a relatively small likelihood of 'black' pixels to occur as clusters for these images; hence, having two 'black' neighbors has little meaning in the labeling of the gray pixel. Thus, a quantitative characterization of the images is needed. The biasing of the classification can then be based on this quantification.

The goal of Chapter 4 is the design of a practicable scheme that links the development of the lattice models in Chapter 3 to the biasing scheme discussed above.

Section 4.2 presents the algorithm to simulate binary images using the $2 \times N$ model (with $h \neq 0$), and Section 4.3 presents a usage of the proposed model in the analysis of multispectral data. In Section 4.4, the model is applied to data for which the performance bounds on conventional classification analysis are known. The improvement in performance through use of the $2 \times N$ model in the analysis justifies the model development. Finally, some results on the analysis of real multispectral data are presented in Section 4.5.

Of the three models considered previously, the $2 \times N$ model is pursued for the following reasons:

- While the $1 \times N$ model is analytically tractable for the case when the external field h is non-zero, and for application to k -ary systems, it models interactions only along a single direction.
- The $3 \times N$ model does not have a solution for the non-zero h case.

In previous literature, models such as the ones proposed here have been presented as Markov chains (cf. [33]). However, this is an inaccurate representation. In the present context, the models can not be fashioned after time-series, and do not have a definition of 'past' and 'future' pixels.

4.2 Model Simulation

This section illustrates the procedure to implement the model to simulate a binary system with a given q and h on a two dimensional lattice of arbitrary size. This is important because the corruption induced in a computer simulation can produce marked deviations from the predicted behavior of the system. While the model is precise, the implementation is, at best, a close approximation.

4.2.1 Monte Carlo simulation

The goal of this section is a demonstration of a technique to output a binary system (black and white image) with the desired values of correlation and magnetization for the corresponding q and h input to the algorithm.

Recall Equation 3.1, the distribution for the lattice system -

$$P_X(X) = \frac{1}{Z} e^{-E(X)}.$$

Also recall that correlation C and magnetization M are macroscopic properties of the binary image computed as averages with respect to the above distribution (cf. Equations 3.3-4). Given the values (desired of the output image) of C and M , the tables of Chapter 3 can be used to look up corresponding values of q and h for the $2 \times N$ model. Thus the target distribution P_X can be identified (cf. Figure 3.7 for the relevant details.)

The theory underlying the procedure is sketched below. The software for the simulation is included in Appendices B, D.

The Monte Carlo method [48] [49] is used in the simulation of the stochastic process, with P_X as the unique equilibrium distribution. The observations on the said process possess the desired values of C and M .

A dynamic Monte Carlo method is used to simulate the stochastic process on the computer starting from an arbitrary initial configuration. The process' attainment of equilibrium is accompanied by the convergence of sample averages of correlation and magnetization, \hat{C} and \hat{M} to the respective P_X -averages C and M . In practice, the opposite tack is taken. Equilibrium is said to have been attained by the process when successive observations of the process return minor changes in readings of \hat{C} and \hat{M} . This approach

has performed adequately in most test-simulations (refer to Section 4.2.5 for experimental results).

For the simulation, the stochastic process is designed to have the Markov property³. It comprises a sequence of random variables X_0, X_1, X_2, \dots , with the sub-script being used to represent the time index. Each of the random variables assumes states from the state space corresponding to the binary $2 \times N$ chain system of Chapter 3.

This stochastic process is uniquely determined by the initial distribution $\mathbf{P}(X_0)$ and the transition probabilities $\mathbf{P}(X_{t+1}|X_t)$. Assume that the process satisfies the following two conditions.

- Irreducibility - $\mathbf{P}(X_{t+1}=y|X_t=x) > 0$ for every x, y in the relevant state space.
- Detailed balance - i.e.

$$P_x(X_t = x)\mathbf{P}(X_{t+1} = y | X_t = x) = P_x(X_t = y)\mathbf{P}(X_{t+1} = x | X_t = y).$$

Under the above two conditions, it can be guaranteed [50] that the stochastic process converges as $t \rightarrow \infty$ to the equilibrium distribution P_x regardless of the initial distribution $\mathbf{P}(X_0)$, and thus $\hat{C} \xrightarrow{t} C$, $\hat{M} \xrightarrow{t} M$. [48] discusses these issues in detail. The pertinent material is extracted from the text and presented here.

Let a_{xy} be probability that the stochastic process sees a transition from state x to state y . It can be shown that the principle of detailed balance is satisfied if the following condition holds.

$$a_{xy} = F\left(\frac{P_x(y)}{P_x(x)}\right), \quad (4.1)$$

where $F(\bullet)$ is any function mapping the non-negative real line to the unit interval, and satisfying

$$\frac{F(z)}{F(1/z)} = z, \quad \forall z \in [0, \infty].$$

that satisfies the property $F(z)=z$. By definition of the distribution P_x ,

$$\frac{P_x(y)}{P_x(x)} = \exp(-E(y) + E(x)) = \exp(-\Delta E). \quad (4.2)$$

Two propositions for the function F are as below:

- Metropolis [51] - $F(z)=\min(1, z)$.
- Hastings [52] - $F(z) = z/(1+z)$.

Irreducibility can be ensured by sweeping through the entire lattice in a random fashion and proposing changes in the system state by altering a single site in the lattice at a time. Thus successive observations on the process differ at most in a single link on the $2 \times N$ chain.

The ideas of the above presentation are the foundation of the algorithm for simulation⁴, presented below.

STEP 1. Assign labels randomly to each element in the lattice.

STEP 2. Construct a $2 \times N$ looped chain through the lattice. The elements comprising the chain may be chosen randomly or selected from a look-up table. (See Figure 4.3 below for an illustration).

STEP 3. For each link in the chain –

STEP 4. Select a new configuration for the link;

STEP 5. Compute the difference in energies⁵ between the old and the new configuration;

STEP 6. Use the Metropolis/Hastings algorithm [48][51][52] to decide on transition.

STEP 7. Assess system for convergence. If not converged, re-do process from STEP 2.

STEP 8. Terminate.

³The Markov property cited above implies that the process is 'memoryless', i.e. $\mathbf{P}(X_{t+1}|X_t, X_{t-1}, \dots) = \mathbf{P}(X_{t+1}|X_t)$ [50]. Though 'Markov chain' is the term by which the process is recognized in literature, this usage is foregone to prevent confusion with the definition of the $2 \times N$ chain in this monograph.

⁴ The above presentation is an aggressive condensation of a deep theory, intended as a background to the algorithm presented here. A rigorous treatment is beyond the scope of this dissertation, and can be accessed in the works of [48] [49] [51] and [52], among others.

⁵ The usage of the term 'energy' as a descriptor of system state is for historical reasons, and reflects the origins of this theory in the statistical mechanics explanation of the thermodynamic behavior of multi-particle systems. In the present context, the energy of a system state will be in the sense of the usage in Chapter 3, as a measure of the coherence in the configuration vis-à-vis the labeling.

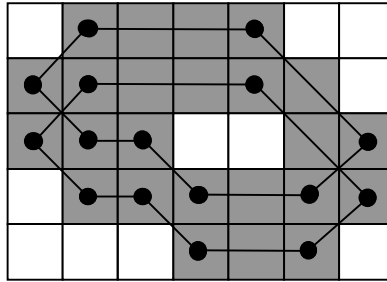


Fig. 4.3: A sample looped 2×12 chain through the given 5×7 lattice system.

Though the method is simple in principle, there are several issues that need to be addressed, namely

- Construction of chain for each of the models.
- Energy computation for a given link configuration.
- Choice of random number generator.

Each of the issues above is assessed individually in the remainder of Section 4.2.

4.2.2 Construction of chain

The simulation calls for the construction of a looped chain through the two-dimensional lattice as in Figure 4.3. The design has to be easily implemented, and yet satisfy the minimum requirements for each of the models.

The designed algorithm selects a number of nodes randomly in the two-dimensional lattice and connects them with straight lines in the order of the selection. The first node in the chain is linked to the last to complete the loop. All elements that are nodes of the chain or lie on the straight lines connecting the nodes become links on the chain. Though not essential in practice, to prevent self intersecting chains, the number of nodes can be restricted to three. The algorithm is sketched in pseudo-code in Figure 4.4, and the software in C programming language is included in Appendices C, E.

```
int chain_path(int row_count, int column_count)
{
N=min(row_count, column_count);
chain = assign_memory(1, row_count*column_count);
link_number=1;
itemp = jtemp = choose_random_number (1, row_count*column_count);
chain[link_number] = jtemp;

/*N nodes are randomly selected in the 2D lattice. the nodes are joined with
straightlines. The chain comprises the nodes + the pixels lying on the lines.
Define temporary variables - jtemp, ktemp as nodes on the chain, itemp as an
element on the line joining jtemp and ktemp*/

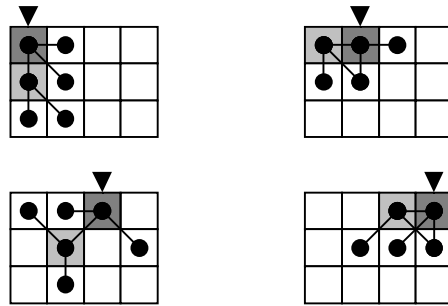
for index = 1 to N-1
  {
    ktemp = choose_random_number(1, row_count*column_count);
    do {
      jtemp=itemp;
      link_number ++;
/*find the slope between the nodes and find next element along the straight line
joining the two nodes*/
      slope = find_slope (jtemp, ktemp);
      itemp = find_next_location(jtemp, slope);
      chain[link_number] = itemp;
    }while (itemp != ktemp);
  }
itemp=jtemp=chain[link_number];
ktemp=chain[1];
/*finish loop by taking the last node in the chain as the first one*/
do {
  jtemp=itemp;
  link_number ++;
  slope = find_slope (jtemp, ktemp);
  itemp = find_next_location(jtemp, slope);
  chain[link_number] = itemp;
}while (itemp != ktemp);
}
```

Fig. 4.4: Pseudocode for generation of chain through two-dimensional lattice.

For the $2 \times N$, and the $3 \times N$ models, the algorithm of Figure 4.4 is modified to generate looped chains running parallel to the side(s) of the chain. For the $2 \times N$ model, the output chain looks like the example shown in Figure 4.3.

While adequate, the above technique can be computationally expensive. Instead of creating a chain through the image, it was realized that a localized representation of the chain would be adequate. Such a representation would enable an implementation in which each element in the lattice could be examined in a raster scan. Experimentally, substantially fewer iterations are needed for convergence. Figure 4.5 illustrates the implementation for the $2 \times N$ model. As suggested by the pointer in the representation, the algorithm proceeds in raster scan down the image. The scheme consists of randomly selecting one of the elements in the eight-element neighborhood of the examined pixel, thus completing the chain-link whose state is considered for change. The next stage requires the identification of the neighbors whose states affect the energy of the configuration. As before, this is done through random selection in the respective eight-element neighborhoods of the two link-pixels. Note that the algorithm requires uniqueness of neighbors for each of the link-pixels, but the four neighbor elements need not be distinct. Refer to Figure 4.6 for the algorithm pseudo-code.

Fig. 4.5: Four steps in the localized representation of a $2 \times N$ chain. The pointer indicates



the element in respect to which the relevant chain links are constructed. The shaded elements represent the elements whose states are considered for alteration as per the Metropolis/Hastings algorithm.

```

pseudo_link(int pointer, int row_count, int column_count)
{
/*Initialize the neighboring pixels of 'pointer'*/
N=E=W=S=NE=NW=SE=SW=0;

/*Examine the eight neighbors of 'pointer' for existence and assign index values to
the corresponding variable- Note that index-value ranges from 1...row_count ×
columncount*/
set_neighbor_indices(pointer, N,E,W,S,NE,NW,SE,SW);

/*Randomly select neighbor of pointer, thus completing the chain-link*/
link_to_pointer =choose_valid_neighbor(N,E,W,S,NE,NW,SE,SW);

/*Select two distinct neighbors of pointer*/
do{
neighbor_to_pointer_EAST=choose_valid_neighbor(N,E,W,S,NE,NW,SE,SW);
}while(neighbor_to_pointer_EAST==link_to_pointer);
do{
neighbor_to_pointer_WEST=choose_valid_neighbor(N,E,W,S,NE,NW,SE,SW);
}while(neighbor_to_pointer_WEST==neighbor_to_pointer_EAST)OR
(neighbor_to_pointer_WEST==link_to_pointer);

/*Select two distinct neighbors of link_to_pointer*/
do{
neighbor_to_link_EAST=choose_valid_neighbor(N,E,W,S,NE,NW,SE,SW);
}while(neighbor_to_link_WEST==index)
do{
neighbor_to_link_WEST=choose_valid_neighbor(N,E,W,S,NE,NW,SE,SW);
}while(neighbor_to_link_WEST==index)OR
(neighbor_to_link_WEST==neighbor_to_link_EAST);
/* At the end of the above steps, the program should have the locations for a pixel,
its neighbor, and their respective 'EAST' and 'WEST' links. Note that the four
'EAST', 'WEST' links need not be distinct. The pseudo-link looks like
neighbor_to_pointer_EAST -- pointer --neighbor_to_pointer_WEST
|
neighbor_to_link_EAST--link_to_pointer--neighbor_to_link_WEST
The metropolis algorithm computes energy difference based on flips to the pixels
located at pointer and link_to_pointer. Consequently, only the bonds of relevance to
the energy difference are shown.*/

```

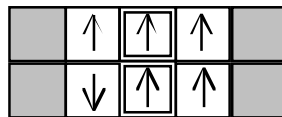
Fig. 4.6: Pseudocode for generation of a pseudo-link to a given element in a finite two-dimensional lattice.

4.2.3 Energy computation and the Metropolis algorithm

The Monte Carlo Markov Chain technique [48] [49] is a general method for the simulation of stochastic processes. While it is hard to simulate statistically independent realizations of the stochastic process, it is possible to achieve an approximation through an iterative procedure. At every iteration, an alternate system-state is proposed, and the proposed state is accepted on the basis of a probability rule [51] [52] applied to the energy difference between the old and the new system configurations. After a suitable number of iterations, the successive system realizations are in accordance to the desired stationary distribution. In this section, the probabilistic transition from one state to the next will be discussed.

The system, in our simulation, is pre-assigned a random configuration. Once a $2 \times N$ chain has been created in the lattice, a given link is chosen, and its configuration is altered. The change in energy of the system, ΔE , is computed (cf. $E(X)$ in Figure 3.7). The computation is illustrated in Figure 4.7 below (the Up-arrow denotes the state of +1, and the Down-arrow denotes the state of -1). Note that the elements whose orientations are altered are enclosed in the double-bordered boxes. The computation shown in Figure 4.7 is in regard to the orientations of only these elements since the configuration of the remainder of the chain is irrelevant to the change in the energy. Correspondingly, the elements outside the scope of interaction with the chosen link are grayed out in the representations.

1. Original state



2. Candidate state

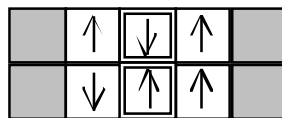


Fig. 4.7: $\Delta E = E_2 - E_1 = -q[(-3) - 3] - h[0 - 2] = 6q + 2h$.

The transition of chain to the candidate state is done with the probability $\min[1, \exp(-\Delta E)]$ [51], or $\exp(-\Delta E)/[1 + \exp(-\Delta E)]$ [52].

4.2.4 Random number generator (RNG)

In any Monte Carlo simulation, the choice of the random number generator (RNG) is critical. A variety of techniques have been proposed in various literature [48] [53], some better than others. The choice of the RNG in the present work is that of a one-step multiplicative congruential linear recurrence generator [49]. This choice is based not only on the appearance of sufficient irregularity (or ‘randomness’) in the output, but also on computational efficiency of the algorithm. The latter point is important from the point of view of implementation, since any Monte Carlo simulation can comprise several thousand runs of the algorithm.

For the one-step multiplicative congruential generator, there are two parameters governing the output, the multiplier a and the period T . For a particular seed V_0 input to the RNG, the output of the algorithm is as below.

$$V_i = aV_{i-1} \pmod{T} \quad \forall i = 1, 2, 3, \dots,$$

$$U_i = \frac{V_i}{T}.$$

U_i 's are the output of the RNG, comprising a set of independent uniform deviates from the unit interval $[0, 1)$. A critical property of this class of RNGs is the periodicity of the output, as determined by T . The size of the period thus governs the design of the RNG. However, the size of the period is limited by the representation of a long integer in the computer. Ignoring the sign bit, it follows that the largest integer capable of representation on most desktop computers is $2^{31}-1$. It has been found that the choice of the primary roots of T [49] for the multiplier a ensures that the RNG attains full period ($=T$). The selection of the primary root a should be such as to induce irregularity in the output. It should be noted that irregularity in a single dimension does not imply irregularity in the output in two or higher dimensions. Hence, though a is frequently chosen to be 16,807, the performance of the thus designed RNG is inferior to that designed with the primary roots 48,271 or 69,261, especially in dimensions higher than

one. This assertion follows from results of the spectral test for randomness [pp. 615, 49] on the RNG.

The RNG used in this work uses the modulus $T=2^{31}-1$, and primary root $a=48,271$.

4.2.5 Experimental results

This section demonstrates the simulation algorithm. For the purpose of illustration, three 'natural' images were obtained from various sources [54] [55] and thresholded (at gray-level 100) to convert the grayscale image to a pure black-white binary form. The correlation and the magnetization for each of the images were estimated, and mapped to q and h via Tables 3.2-3 for the $2 \times N$ model. These values of q and h were input to the algorithm to generate the respective simulations. The original images and their simulations are presented overleaf. For a quantitative evaluation of the performance, the correlation and the magnetization of the original and the generated images are presented in Table 4.1.

As may be apparent from the tabulated results, and the subjective evaluations of the figures on the next page, the algorithm works reasonably well for simulating 'PlasticBubs' and 'BrodatzBeachSand'. Note that the algorithm is not intended to replicate the input images, but rather induce the input correlation and magnetization in the simulation. Furthermore, the subspace of images possessing the given values of C and M is quite large. The output of the algorithm is statistically unlikely to reproduce the parent image.

Table 4.1

Measurements of correlation (C) and magnetization (M) for sample binary images, and for their respective simulations.

Image Name	Original image		q	h	Simulated image	
	C	M			C	M
'PlasticBubs'	0.699	-0.208	0.575	-0.020	0.685	-0.221
'NaturalTurbulence'	0.954	-0.100	0.892	-0.005	0.614	-0.542
'BrodatzBeachSand'	0.880	-0.703	0.690	-0.068	0.884	-0.817

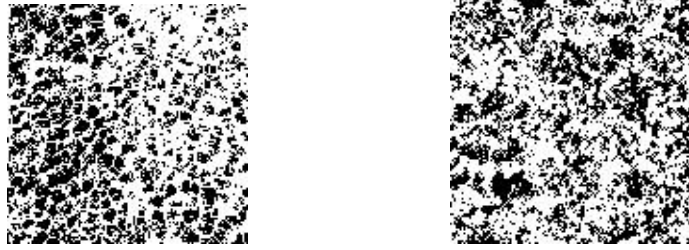


Fig. 4.8: Image 'PlasticBubs' on the left. Simulated image on the right.



Fig. 4.9: Image 'NaturalTurbulence' on the left. Simulated image on the right.

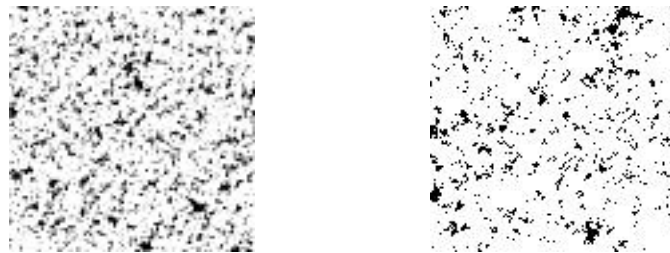


Fig. 4.10: Image 'BrodatzBeachSand' on the left. Simulated image on the right.

As may be noted, the results of the simulations for 'Plasticbubs' and 'BrodatzBeachSand' seem in greater visual correspondence to the original images than the result for 'Naturalturbulence'. This disparity is also evident as the differences among the correlation and magnetization estimates for the set of original and simulated 'NaturalTurbulence' images. An explanation for these dissimilarities is proposed in Appendix A.8.

To summarize, Section 4.2 demonstrates that it is possible to quantify subjective assessments of binary images, and subsequently replicate the macroscopic properties (quantitatively) in images generated through the Monte Carlo method.

4.3 Application to Multispectral Analysis

This section presents a technique for the incorporation of the proposed $2 \times N$ looped chain model in multispectral data analysis. Section 4.3.1 will present the problem and the various assumptions generally used in the analysis of multispectral (remote sensing) data. Section 4.3.2 will present the algorithm used in Sections 4.4-5 for the segmentation of multispectral/multidimensional data. The software for the implementation is included in Appendix C at the end of the text.

4.3.1 Problem description and implementation assumptions

Satellite sensors gather data by measuring the energy reflected off the Earth's surface [1]. The data is spatially quantized into pixels, and radiometrically quantized into discrete 'brightness' levels at each of several wavelengths. Remote sensing data is thus different from the common notion of an image in that each of the sampling wavelengths can be represented as a distinct image (in the visual sense). In this respect, the pixel in the context of remote sensing data analysis is a vector of length equal to that of the number of spectral samplings. The task of the analyst is to identify the land cover associated to a given pixel.

For the remainder of Section 4.3, the following notation will be used in regard to the data. The data will be denoted $\mathbf{Y}=(\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_N)$, with N being the total number of pixels in the image. Each \mathbf{y}_j will denote a vector of length p (also termed the dimensionality of the data). To each \mathbf{y}_j we seek to assign a class-label x_j , taking values from a finite state space $\Lambda=\{-1, 1\}$, such that the class assignment to \mathbf{Y} will be $(x_1, \dots, x_N)=\mathbf{X} \in \Lambda^N$. The ensuing discussion will incorporate the $2 \times N$ chain model of Chapter 3 to model the distribution of the system state \mathbf{X} .

A useful assumption for analysis is that the \mathbf{y}_j are multivariate normal random variables [1] [2], with distribution parameters dependent on the respective label assignment x_j . The analysis is best illustrated through the following example.

Let $\mathbf{X}_{\text{old}}=(x_1, \dots, x_m=1, x_{m+1}=1, \dots, x_N)$ and $\mathbf{X}_{\text{new}}=(x_1, \dots, x_m=1, x_{m+1}=-1, \dots, x_N)$, with the task being to determine which of the two label assignments is the more accurate labeling, given the multivariate data \mathbf{Y} on the scene. (x_m, x_{m+1}) may be viewed as a link in the $2 \times N$ model, in the sense of Section 4.2. The goal of the analysis is the determination

of the scene classification/configuration \mathbf{X} that is the best fit to the spectral data \mathbf{Y} . Bayesian analysis [56] suggests that the decision should be made on the basis of the following decision rule -

$$P(\mathbf{X}_{\text{old}}, \mathbf{Y}) \underset{\mathbf{X}_{\text{new}}}{\overset{\mathbf{X}_{\text{old}}}{>}} P(\mathbf{X}_{\text{new}}, \mathbf{Y}). \quad (4.3)$$

Here, $P(\mathbf{X}_{\{\text{old, new}\}}, \mathbf{Y})^6$ represents the joint probability of occurrence of the spectral data \mathbf{Y} , and the corresponding classification \mathbf{X} (the sub-script added as per requirement). The decision rule in expression 4.3 chooses the configuration that maximizes the respective probability of occurrence. The joint probabilities in expression 4.3 can be represented in terms of conditional probability distributions of $\mathbf{Y}|\mathbf{X}$ [57] (sub-script added as per requirement). The representation of the associated density functions is as $p(\mathbf{Y}|\mathbf{X}_{\{\text{old, new}\}})$.

Simplification of expression 4.3 yields

$$\begin{aligned} P(\mathbf{X}_{\text{old}})p(\mathbf{Y}|\mathbf{X}_{\text{old}}) &\underset{\mathbf{X}_{\text{new}}}{\overset{\mathbf{X}_{\text{old}}}{>}} P(\mathbf{X}_{\text{new}})p(\mathbf{Y}|\mathbf{X}_{\text{new}}) \\ \prod_{j=1}^N P(\mathbf{X}_{\text{old}})p(\mathbf{y}_j | x_j \in \mathbf{X}_{\text{old}}) &\underset{\mathbf{X}_{\text{new}}}{\overset{\mathbf{X}_{\text{old}}}{>}} \prod_{j=1}^N P(\mathbf{X}_{\text{new}})p(\mathbf{y}_j | x_j \in \mathbf{X}_{\text{new}}) \\ P(\mathbf{X}_{\text{old}})p(\mathbf{y}_m | x_m = 1)p(\mathbf{y}_{m+1} | x_{m+1} = 1) &\underset{\mathbf{X}_{\text{new}}}{\overset{\mathbf{X}_{\text{old}}}{>}} P(\mathbf{X}_{\text{new}})p(\mathbf{y}_m | x_m = 1)p(\mathbf{y}_{m+1} | x_{m+1} = -1) \end{aligned}$$

The above simplifications are based on the following assumptions -

- Given the labeling $\mathbf{X}_{\{\text{old, new}\}}$, the \mathbf{y}_j are multivariate Gaussian random variables.
- Given the labeling $\mathbf{X}_{\{\text{old, new}\}}$, the \mathbf{y}_j are statistically independent.
- The system state $\mathbf{X}_{\{\text{old, new}\}}$ has a Gibbsian distribution [16][9], a function of q and h . Thus, as in Equation 2.1

$$P(\mathbf{X}) = \frac{1}{Z} \exp(-E(\mathbf{X})).$$

Taking the natural logarithm on both sides of the above expression, the decision rule reduces to

⁶ For ease of representation, no notational distinction is made between the random variable and its observation.

$$-E(\mathbf{X}_{\text{old}}) + E(\mathbf{X}_{\text{new}}) + \ln \left[\frac{P(\mathbf{y}_m | x_m = 1)P(\mathbf{y}_{m+1} | x_{m+1} = 1)}{P(\mathbf{y}_m | x_m = 1)P(\mathbf{y}_{m+1} | x_{m+1} = -1)} \right] \begin{matrix} \mathbf{X}_{\text{old}} \\ > \\ \mathbf{X}_{\text{new}} \end{matrix} < 0,$$

Using ΔE to represent the difference in the 'energy' functions above, and $r(\bullet)$ for the remainder of the expression, the above decision rule is re-written as

$$\Delta E + r(\mathbf{y}_m, \mathbf{y}_{m+1}) \begin{matrix} \mathbf{X}_{\text{old}} \\ > \\ \mathbf{X}_{\text{new}} \end{matrix} < 0. \quad (4.4)$$

If q and h are known, ΔE can be computed by observations of the states of the neighbors to x_m and x_{m+1} (cf. Figure 3.7, and Section 4.2.2 for details). The function $r(\bullet)$ can be computed via the Gaussian assumption on the spectral data. Briefly, the classification analysis of spectral data using the schemes developed so far is as below:

- The binary image system (or scene-classification) is scanned in raster fashion. A $2 \times N$ chain is simulated on the initial classification map for the data, as in the illustration of Figure 4.5.
- The correlation C and magnetization M are estimated, and Tables 3.2-3 used to estimate the corresponding values of q and h for the system.
- An alternate configuration for the classification map is suggested by changing one or more element states in the analyzed pixel pair.
- The two choices of configurations are compared to obtain ΔE using the values of q and h estimated earlier.
- The decision rule suggested by expression 4.4 is used to decide the best configuration.
- A critical deviation from the rule however, is that the probabilistic transition rule of [51] [52] is incorporated. This is required because the state space for all classification maps $\{\mathbf{X}\}$ is enormous, and an exhaustive search for the configuration that maximizes the probability $P(\mathbf{X}, \mathbf{Y})$ is impractical. The Monte Carlo method [48] [49] is invoked, and used in the manner of [14] [58]. It is believed that successive iterations in the implementation lead to the discovery of a (near-)optimal classification map for the system. [51] suggest

that the change in link configuration be made with probability $\min(1, \exp[-(\Delta E + r(\mathbf{y}_m, \mathbf{y}_{m+1}))])$.

The next section details the algorithm for implementation.

4.3.2 Algorithm for implementation

The estimation of distribution parameters (mean and covariance) is based on previous work by [18] and the Expectation Maximization (EM) algorithm⁷ [59] [60]. The details of that work will not be included here, but can be found as comments to the code for the implementation included in Appendix C.

- STEP 1. Input multispectral data \mathbf{Y} , binary system labeling \mathbf{X} . \mathbf{Y} comprises data drawn from either one of two multivariate normal distributions.
- STEP 2. Using the class assignment to each element in \mathbf{Y} , estimate mean and covariance for each of the said normal distributions using the EM algorithm. These are used to compute the $r(\bullet)$ function of expression 4.4.
- STEP 3. Measure the inter-pixel correlation, and magnetization of the system from \mathbf{X} . Estimate q, h using Table 3.2.
- STEP 4. Select pixels in raster order. For each pixel:
 - STEP 5. Construct a chain link as per the scheme of Section 4.2.2.
 - STEP 6. Select a pair configuration different from the present one.
 - STEP 7. Compute $\Delta E + r(\bullet)$ as the link energy differential (refer expression 4.4).
 - STEP 8. If the energy differential favors the new configuration (of lower energy), then enforce the new labeling on the link pair. Otherwise, make the transition with a probability $\exp[-(\Delta E + r(\bullet))]$ [48] [51].
- STEP 9. Repeat from STEP 2 until convergence is observed.

The development of the $2 \times N$ model is justified by STEP 3. However, the above algorithm is of seemingly limited utility, given that most systems of interest in remote

⁷Statistical analysis of remote sensing data centers on the accurate estimation of the conditional distributions $\mathbf{Y}|\mathbf{X}$. Under the Gaussian assumption, stated earlier, the estimation requires the computation of sample means, covariances for input to the analysis. The EM algorithm is used here in the manner of [60] [90] to improve the estimation accuracy.

sensing data analysis are classified as multipolar (more than two classes) thematic maps. Given data \bar{Y} with a multipolar labeling \bar{X} , analysis may proceed in one of two ways -

- It could be surmised that one of the scene classes used in the labeling \bar{X} , possesses sub-classes, the knowledge of which would be informative (see the experiment on the HYDICE data in Section 4.5). In this case, all data labeled the class of interest is extracted as Y . This data can be labeled to one of two classes using a simple segmentation algorithm such as ISODATA [61] to provide a bipolar classification map X .
- In the other case, it could be observed that due to spectral similarity among the spectral signatures for two of the scene classes, there is significant confusion in the classification of the corresponding data. In this scenario, all data classified as either of the two classes can be extracted as Y , and the respective labelings retained as X for input to the algorithm. The experiment detailed in Section 4.5 on the D.C. flightline data uses this scheme to separate GRASS from TREES.

Hence, given spectral data \bar{Y} and an associated multipolar labeling \bar{X} , $Y \subset \bar{Y}$ and the corresponding bipolar labeling $X \subset \bar{X}$ are obtained. These Y and X can now be used as input to the algorithm

4.4 Rationale for Usage – An Experimental Evidence

In this Section, the justification for the spatial-spectral scheme is presented via an experiment. In general, in the analysis of remote sensing data, it is hard to quantify improvements in accuracy in the absence of ground truth. To verify that the proposed technique is of practical worth, the following experiment was conducted.

4.4.1 Experimental design

- Fix p , the dimension of the generated multivariate data.
- Fix η as a vector of size p , and all elements equal to 100.
- Fix $\mathbf{I}_{p \times p}$ as the $p \times p$ identity matrix.
- Generate 10,000 data as a sample from a multivariate Gaussian process [62] with mean η , and covariance $100 \bullet \mathbf{I}_{p \times p}$. These data will subsequently be referred to as D_1 .
- Distribute the generated data on the top half of a 400×500 lattice - i.e. to each of the 10,000 elements in D_1 assign a distinct location on the top half of the lattice.
- Generate 10,000 data as a sample from a multivariate Gaussian process [62] with mean η , and covariance $400 \bullet \mathbf{I}_{p \times p}$. These data will subsequently be referred to as D_2 .
- Distribute D_2 on the bottom half of the 400×500 lattice.

If D_1 and D_2 are labeled classes I and II respectively, a visual representation of the class distribution on the lattice is the black and white image in Figure 4.11. This classification map however, is hidden and must be inferred from the p dimensional data on the image.

A representation of the data for $p=2$ is shown as a two color scatter plot in Figure 4.12. Note that D_1 , on account of the smaller variance per channel, has a lesser spread than D_2 across the plot.

The above presentation is an idealized representation of most applications in remote sensing data analysis. However, due to knowledge of the underlying class-conditioned distributions in this experiment, it is possible to compute the performance bound for conventional analysis of the data.

Since the two scene classes are equally likely (by construction), the Bayes decision rule [56] for classification of the data is obtained –

$$r(\mathbf{y}) = \frac{3}{800} (\mathbf{y} - \boldsymbol{\eta})^t (\mathbf{y} - \boldsymbol{\eta}) + \frac{1}{2} \ln \frac{1}{16} \begin{matrix} II \\ > \\ I \end{matrix} < 0.$$

Here, \mathbf{y} is the p -dimensioned data on the element to be classified, and $\boldsymbol{\eta}$ is the mean, as designed for the experiment. Statistically, the Bayes rule designed above is the best possible decision rule for the separation of the data into the two underlying classes. The corresponding errors in classification can be computed as

$$P_{err}^I = \Pr(1.5\chi_p^2 + p1 n 2 > 0), \tag{4.5}$$

$$P_{err}^{II} = \Pr(0.375\chi_p^2 + p1 n 2 < 0).$$

$$P_{err}^{avg} = \frac{1}{2} (P_{err}^I + P_{err}^{II}). \tag{4.6}$$

χ_p^2 is the Chi-squared random variable, with p degrees of freedom.

In Equations 4.5-6 P_{err}^I is the proportion of D_1 that were wrongly identified in the mixture of D_1 and D_2 . Likewise, P_{err}^{II} is the proportion of D_2 in the mixture that were misclassified.

P_{err}^{avg} is the Bayes error in the analysis, averaged from P_{err}^I and P_{err}^{II} , as in Equation 4.6.

These errors can be analytically computed. Table 4.2 lists the Bayes errors over a range of data-dimensions p . Note that the error decreases with increase in p .

Next, fifteen sets of test data are generated for each of several selections of p in the range 2-15. The analysis scheme of Section 4.3 using the $2 \times N$ model is implemented. Since the underlying classification map is known (by design), the error in classification can be obtained for each analysis output. For every selection of p , the fifteen values of obtained classification-error are averaged and tabulated in Table 4.2 under 'Experimental error'.

The results of Table 4.2 are also plotted in Figure 4.13.

Table 4.2

Tabulation of classification performance of proposed algorithm on simulated data, with corresponding values of theoretical Bayes error listed alongside for comparison.

p	Bayes error	Experimental error (averaged over 15 test runs)
2	0.264	0.168
3	0.214	0.118
4	0.176	0.087
5	0.148	
6	0.124	0.051
7	0.106	
8	0.090	0.034
9	0.077	
10	0.066	0.023
11	0.057	
12	0.049	0.016
13	0.043	
14	0.037	0.011
15	0.032	

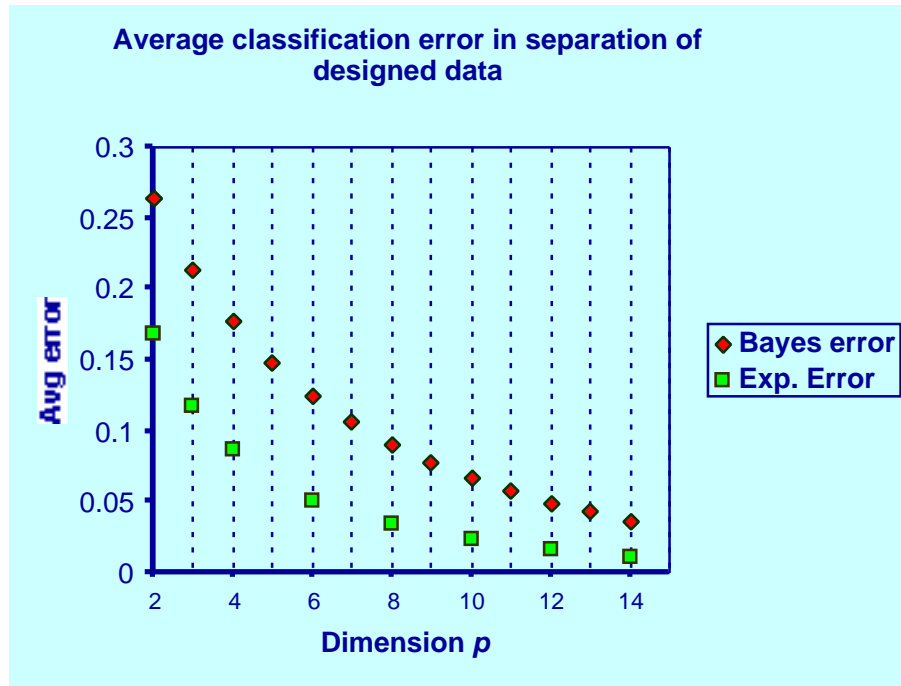


Fig. 4.13: Classification performance on experimental data. 'Bayes error' plots the theoretical best-case performance of analysis without use of spatial information. 'Experimental error' plots the results for the classification performance, averaged over 15 test runs for select values of p , for analysis using the $2 \times N$ lattice model.

4.4.2 Results interpretation

Theoretically, it is not possible to do better than the result predicted by the Bayes decision rule of Equation 4.3 - if each element is classified solely on the basis of the associated multispectral data. However, from Figure 4.13 it is evident that the use of the $2 \times N$ model is an improvement over conventional classification analysis.

It can be concluded that the improvement is a result of the spatial adjacency information incorporated with the $2 \times N$ model. Logically, the decision should favor classification that is consistent with the labeling of the pixel's neighbors; i.e. if a given pixel has neighboring pixels that are labeled (say) Class I, there is a high likelihood the identified pixel should be labeled Class I as well. The decision rule should thus be biased in favor of labeling a pixel the same as its neighbor(s). This is precisely the notion of the prior in Bayesian analysis and the $2 \times N$ model is used to compute the magnitude of the bias. The discussion in this section attempts to estimate the improvement in the classification as a function of the pixel correlation in the image.

For ease of notation, the following presentation fixes the dimensionality of the data as $p=2$. Figure 4.14 plots the probability density functions for each of the operands in the decision rules of Equations 4.5. Note that -

- f_{r1} plots the density function for $r_1=1.5\chi_2^2+ 2\ln 2$.
- f_{r2} plots the density function for $r_2=0.375\chi_2^2+ 2\ln 2$.
- The density function plots intersect at the decision threshold zero.
- The area under f_{r2} to the left of the decision threshold is $P_{err}^{II}(=0.371)$.
- The area under f_{r1} to the right of the threshold is $P_{err}^I(=0.156)$.
- The 'Bayes error' entry in Table 4.2 for $p=2$ is the average of P_{err}^I and P_{err}^{II} ($=0.264$).

Probability densities for decision functions
in Equations 4.5 for $p=2$.

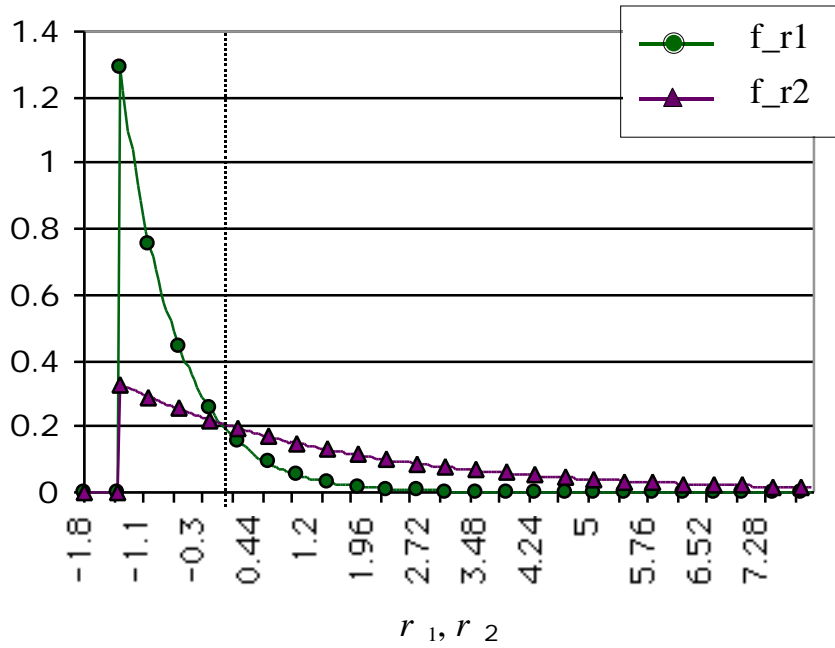


Fig. 4.14: Plots for the probability density functions for the operands of the decision rules in Equations 4.5, for $p=2$.

Recall the Bayes decision rule for separating the designed data.

$$r(\mathbf{y}) = \frac{3}{800} (\mathbf{y} - \boldsymbol{\eta})' (\mathbf{y} - \boldsymbol{\eta}) + \frac{1}{2} \ln \frac{1}{16} \frac{II}{I} > 0$$

The incorporation of the spatial model biases the decision rule⁸ as below.

$$r(\mathbf{y}) = \pm 2q + \frac{3}{800} (\mathbf{y} - \boldsymbol{\eta})' (\mathbf{y} - \boldsymbol{\eta}) + \frac{1}{2} \ln \frac{1}{16} \frac{II}{I} > 0 \quad (4.7)$$

The sign of the bias depends on the label of the pixel's neighbor. The magnitude of the bias in the decision rule reflects the amount the computed statistic should be altered to incorporate information on the examined pixel's neighborhood. For instance, if the two neighbors of the examined element are labeled as Class II (belonging to D_2), the decision rule will be biased by $+2q$ favoring a classification as Class II. Expression 4.7 is however an idealized case in which both neighbors of the analyzed element are labeled the same.

For the given data, the q for the black and white image of Figure 4.11 is 1.44 (cf. Appendix A.7). The effect on the corresponding class conditioned distributions is a lateral shift, of magnitude $2q$, away from the decision threshold, zero.

Note that the direction of the shift depends on the class assignment to the neighbors of the examined pixel. For instance, in the ideal case, any pixel in the top half of the image lattice will have neighbors classified as 'white' (Figure 4.11). However, in practice, the classification map can be corrupted and mislabeled data can lead to an inaccurate bias. In other words, if the said pixel, in the top half of the lattice, has neighbors (mis-)classified as 'black', the decision rule will be biased incorrectly in labeling the analyzed data-element as 'black'. If the neighborhood is an inaccurate bias to the pixel's identification, the shift will corrupt the classification significantly. This point will be illustrated with the help of Figures 4.15-16. Recall that χ_2^2 represents a random variable with the Chi-squared distribution, with two degrees of freedom.

Some comments on Figure 4.15 are listed below.

⁸ The bias corresponds to the ΔE term as presented in expression 4.4. In this case, the neighborhood of the examined element is assumed to be labeled the same, and the total bias is $+2q$ (refer to Figure 4.7 for an example of ΔE computation). Note also, that h is assumed to be zero in this case, on account of equal distribution of the two classes.

- The plots show density function plots for $1.5\chi_2^2 + 2\ln 2$, $1.5\chi_2^2 + 2\ln 2 + 2q$, and $1.5\chi_2^2 + 2\ln 2 - 2q$, according to the bias applied to the decision rule.
- The error in classification of D_1 for each of the biasing schemes is the area under the curve to the right of the decision threshold (=0).
- Clearly, the best performance is observed for the bias $-2q$, which, as per expression 4.7, reflects the neighborhood of the examined element being populated by Class I data.
- If the neighborhood is incorrectly labeled, the penalty of the wrong bias is severe - note the error under the density function for $1.5\chi_2^2 + 2\ln 2 + 2q$ to the right of the decision threshold.

Probability densities for decision functions $r(\mathbf{y})$ to identify Class I data ($p=2$).

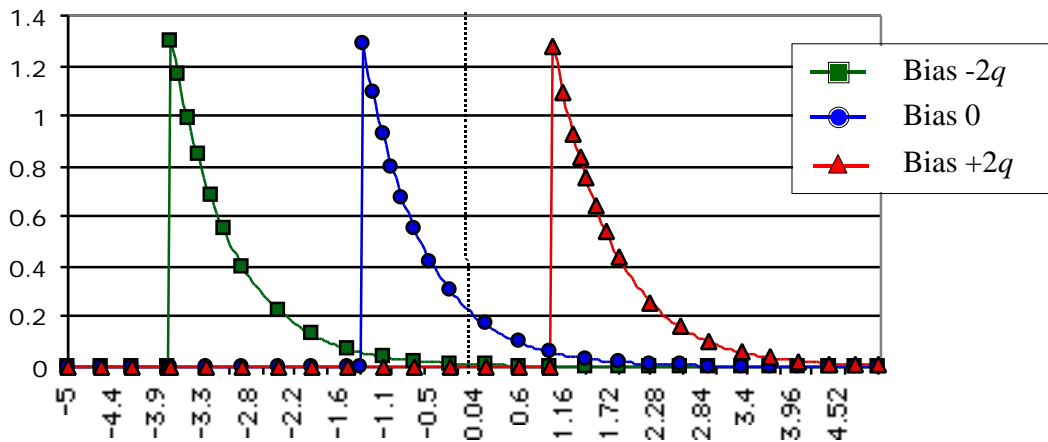


Fig. 4.15: Density function plots for $1.5\chi_2^2 + 2\ln 2 + \{0, -2q, 2q\}$. Note that the area under the curves to the right of the decision threshold (shown as the vertical line) corresponds to the respective errors in classification of the data D_1 .

Some comments on Figure 4.16 are as below.

The plots show density function plots for $0.375\chi_2^2 + 2\ln 2$, $0.375\chi_2^2 + 2\ln 2 + 2q$, and $0.375\chi_2^2 + 2\ln 2 - 2q$, according to the bias applied to the decision rule.

The error in classification of D_2 for each of the biasing schemes is the area under the curve to the left of the decision threshold (=0).

Clearly, the best performance is observed for the bias $+2q$, which, as per Equation 4.7, reflects the neighborhood of the examined element being populated by Class II data.

If the neighborhood is incorrectly labeled, the penalty of the wrong bias is severe - note the error under the density function for $0.375\chi_2^2 + 2\ln 2 - 2q$ to the left of the decision threshold.

Probability densities for decision functions $r(\mathbf{y})$ to identify Class II data ($p=2$).

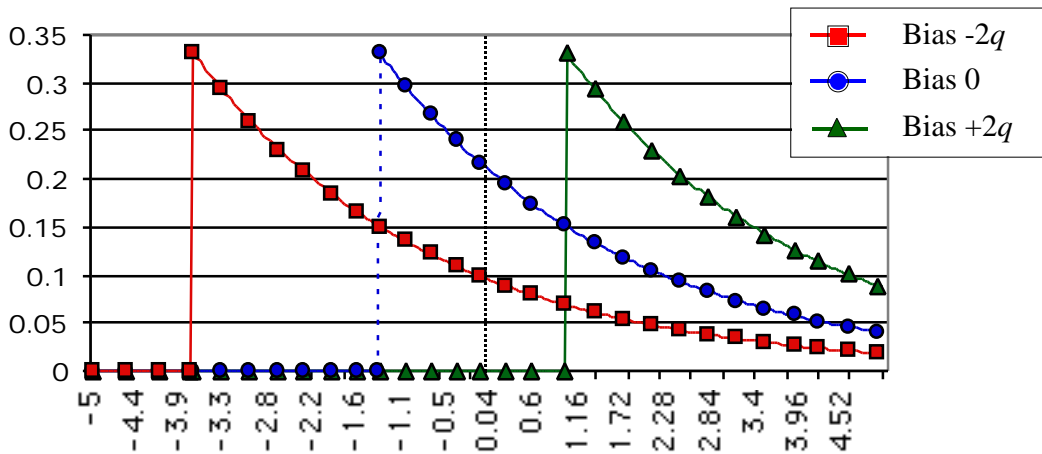


Fig. 4.16: Density function plots for $0.375\chi_2^2 + 2\ln 2 + \{0, -2q, 2q\}$. Note that the area under the curves to the left of the decision threshold (shown as the vertical line) corresponds to the respective error in classification of the data D_2 .

As has been pointed out, there is a heavy penalty for an incorrect bias. In general, it is believed that a neighborhood is strongly representative of the elemental classification. Additionally, if the correlation in the image is small, i.e. the image is highly granular and the possibilities of incorrect biasing is high, then the q is small; thus the bias-induced error is small as well.

The remainder of this section attempts an analytical modeling of the performance of the proposed algorithm, on the designed data, as listed in Table 4.2.

Figures 4.15-16 and the associated explanations have demonstrated that classification error over the designed data is a function of the bias applied to the decision rule. The proposed algorithm incorporates spatial information on the classification map and thus improves on the conventional analysis (as indicated by the entries under Bayes error in Table 4.2). However, it is also evident from Figures 4.15-16 that an incorrect bias can severely deteriorate performance. In general, the neighborhood labeling is a correct indicator of a pixel's classification; and thus the overall performance of the proposed scheme is improved. In modeling the classification accuracy of the proposed algorithm, the incorrect biasing has to be incorporated into the calculations. Consider the classification map below.



Fig. 4.17: A sample classification output of the proposed algorithm implemented on the designed data.

The classification error in the output above, manifest as 'speckle', has a direct role in classification performance of the algorithm. The speckle errors in the classification map induce incorrect biases that lead to further deterioration in performance.

Consider Table 4.3, presented below, in the context of the fifteen test runs of the proposed algorithm on the designed data for different values of dimensionality p .

- The dimensionality of the designed data for each set of experiments is entered under p .
- Given the fifteen sets of data available for each selected p , the listed values of scene correlation (C) are averaged over the observed correlation-values of the respective outputs.
- Table 3.1 is used to estimate q mapped by the C obtained as above (h is assumed to be zero).
- The estimated values of q are used to bias the relevant decision functions –

$$r_1=1.5\chi_p^2+p \ln 2, \text{ and } r_2=0.375\chi_p^2+p \ln 2$$
- Errors in classifying D_1 using the decision rule r_1 using the biases from $\{0, 2q, -2q\}$ are entered under the corresponding columns for P_{err}^I .
- Errors in classifying D_2 using the decision rule r_2 using the biases from $\{0, 2q, -2q\}$ are entered under the corresponding columns for P_{err}^{II} .

Table 4.3
 Listing of experimental readings of scene correlation C (each averaged over fifteen iterations), respective estimates of q , and classification errors for different biasing schemes, for select values of p .

p	C	q	Unbiased($q=0$)		P_{err}^I		P_{err}^{II}	
			P_{err}^I	P_{err}^{II}	Bias= $-2q$	Bias= $+2q$	Bias= $-2q$	Bias= $+2q$
2	0.765	0.652	0.1564	0.3711	0.0274	0.8866	0.5932	0.0297
3	0.822	0.716	0.1352	0.2920	0.0246	0.6263	0.4963	0.0675
4	0.855	0.760	0.1159	0.2370	0.0218	0.4992	0.4197	0.0670
6	0.905	0.842	0.0852	0.1636	0.0161	0.3573	0.3099	0.0515
8	0.935	0.910	0.0631	0.1169	0.0117	0.2682	0.2334	0.0377
10	0.955	0.973	0.0471	0.0852	0.0085	0.2065	0.1784	0.0274
12	0.968	1.031	0.0354	0.0629	0.0061	0.1611	0.1375	0.0199
14	0.975	1.084	0.0267	0.0469	0.0044	0.1265	0.1066	0.0146

Given the nature of the (hidden) ideal classification map of the designed data, the performance of the algorithm can be adequately modeled by considering a vertical two-pixel wide strip representative of the image. Consider Figure 4.18 for the ideal

- Every misclassification results in two flip-counts.
- Thus we have

$$\text{bad_ratio} = \frac{1}{2} \left(\frac{\text{flip_count}}{2 \cdot 99 - 1} \right).$$

- 'bad-ratio' is the proportion in the 98% of the pixels that receive bias, that have 'bad' neighborhoods on account of misclassifications and receive biases in the wrong direction. good-ratio=1-bad-ratio, and its definition follows a similar logic.
- Thus estimated P_{err}^I is calculated as

$$0.98 \cdot (\text{good-ratio} \cdot P_{err}^I \text{ with good bias} + \text{bad-ratio} \cdot P_{err}^{II} \text{ with bad bias}) + 0.02 \cdot P_{err}^I \text{ with no bias}.$$

The values of P_{err}^I for the respective biases can be read from Table 4.3.

- P_{err}^{II} can be estimated using an argument parallel to the above.

Consider an example for $p=2$. The observed correlation (averaged over 15 test runs) is 0.765. The number of transitions, or the flip-count', is estimated as 47, and the corresponding bad_ratio is approximately 11.7%. It is estimated that in identifying 98% of the Class I data (corresponding to the non-transition pixels), the bias is good for approximately $(100-11.7)=88.3\%$, and bad for the remaining 11.7%. 2% of the Class I data receive no bias at all.

Then the entry for $p=2$ for the P_{err}^I , using the values from Table 4.3, can be recalculated as

$$0.98 \cdot (0.883 \cdot 0.0274 + 0.117 \cdot 0.8866) + 0.02 \cdot 0.1564 \approx 0.1283.$$

The remainder of Table 4.4 is computed in a similar fashion.

Admittedly, the development above is a conjecture. However, the algorithm performance does depend on the biasing scheme. The magnitude and sign of the biasing depends on the classification output itself. The above heuristics attempt to relate the accuracy with the correlation observed for the output image. The results of Table 4.4 have been plotted in Figure 4.19, and appear consistent with the experimental observations on classification performance.

Table 4.4
 Predicted error in separation of the designed data, as a function of observed scene correlation C .

p	C	'flip' count	Error prediction		
			P_{err}^I	P_{err}^{II}	Avg.
2	0.765	47	0.1283	0.101	0.1149
3	0.822	35	0.0777	0.1082	0.0931
4	0.855	29	0.0569	0.095	0.0760
6	0.905	19	0.0327	0.0653	0.0490
8	0.935	13	0.0204	0.0452	0.0328
10	0.955	9	0.0132	0.0316	0.0224
12	0.968	6	0.0086	0.0223	0.0155
14	0.975	5	0.0061	0.0162	0.0111

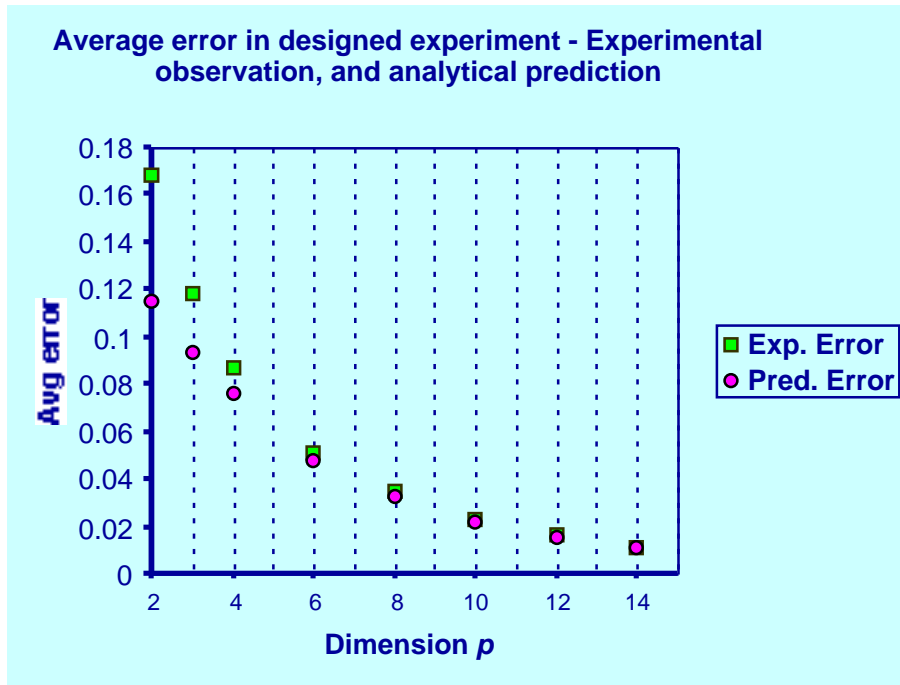


Fig. 4.19: Plot of classification errors in separation of designed data - experimental values and analytical predictions.

4.5 Experiments on Remote Sensing Data

The scheme is implemented on two sets of data. In this section, emphasis will be laid on illustration of the concept. Part Two of this dissertation comprises a comprehensive analysis of one of the datasets (the D.C. flightline) presented here, and the results shown here, will be presented again with more procedural detail, and with a quantitative assessment.

4.5.1 DC flightline data

The data was collected for a flightline over the Washington D.C. mall. The HYDICE scanner was used, and spectral data was collected over 210 channels (0.4-2.4 μm .). The data was rectified at the School of Civil Engineering, Purdue University before being made available for research. The rectified data comprised 1,310 rows and 265 columns of (=347,150 data elements). A three color representation of the data is shown in Figure 4.20. Training data was compiled on the data [63], as in Figure 4.21 and maximum likelihood classification was carried out. The classification yielded a result that had confusion among classes GRASS (or LAWN) and TREES, as observed in Figure 4.22. Since the proposed algorithm operates only on a binary system, the implementation isolated the data with the target class-labels (LAWN and TREES) for segmentation as in Figure 4.23. The segmentation results are shown in Figure 4.24. The text output of the program is included as Figure 4.25 to illustrate the convergence of the algorithm.