8-1-1999

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

Varun Madhok
Purdue University School of Electrical and Computer Engineering

David Landgrebe
Purdue University School of Electrical and Computer Engineering

SPECIAL-SPATIAL ANALYSIS OF REMOTE SENSING DATA: AN IMAGE MODEL AND A PROCEDURAL DESIGN

VARUN MADHOK
DAVID LANDGREBE

TR-ECE 99-10
JULY 1999

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

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 assumptions *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 forperjormance 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, tht: 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 developecl 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 homogeneity[1] (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

'Visually, 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

$$P_X(X) = \frac{1}{Z} e^{-\mathrm{E}(X)}, \tag{2.1}$$

where Z is a normalizing constant known as the partition function, and $\mathrm{E}(\bullet)$ 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 $\mathrm{E}(\bullet)$. :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 $\boldsymbol{A}$ (={-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.
- $\boldsymbol{q}$: 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(•): Given an observation of the system X, the energy of the configuration is measured as E(X). E(•) 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

$$P_X(\mathrm{X}) = \frac{1}{Z} e^{-\mathrm{E}(X)} \tag{3.1}$$

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

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

The summation in Equation 3.2 is over all possible realizations of the system. The issue of the energy function E(•) 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

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

where **A** is the transfer matrix representative of the system model, and Tr(•) is the trace function[2] 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 **A.** If $\{l_i\}$ is the set of eigenvalues of **A** then

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

Later development will show that **A** is symmetric for the models considered. Consequently, the largest (principal) eigenvalue of **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(•) 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

$$C = \frac{1}{N} \sum_{\{X\}} \frac{d\mathrm{E}(X)}{dq} P_X(X) = \frac{1}{N} \frac{d \log Z}{dq} . \tag{3.3}$$

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

$$M = \frac{1}{N} \sum_{\{X\}} \frac{d\mathrm{E}(X)}{dh} P_X(X) = \frac{1}{N} \frac{d \log Z}{dh} . \tag{3.4}$$

The system model is thus the map

$$\mathrm{f} : (q,h) \rightarrow (C,M) . \tag{3.5}$$

---

[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 Nth 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 $i=1,\ldots,$ 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.

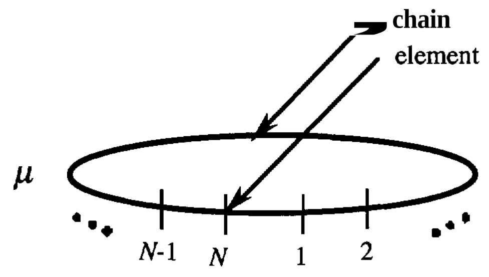


Fig. 3.1: The $1\times N$ lattice model

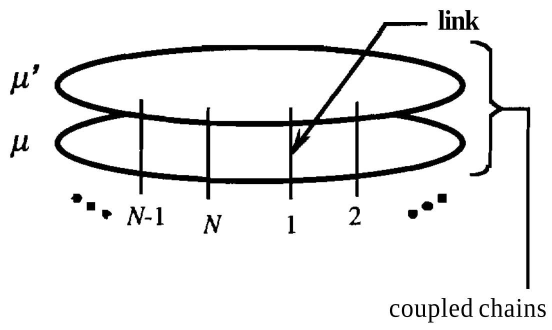


Fig. 3.2: The 2×N lattice model

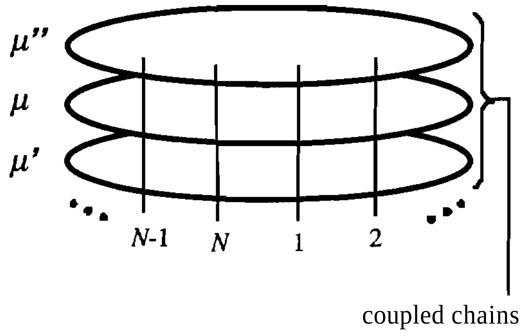


Fig. 3.3: The 3×N lattice model

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×N lattice model. Unfortunately, a similar analysis for the 3×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.

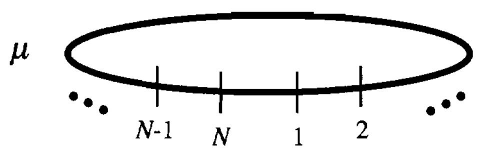
  The interaction among the elements is designed for each model, based on q (and h for Figure 3.7).

  The energy function $E(X)$ is formulated for each model.
- The corresponding transfer matrix A is found. (Recall that $Z=\mathrm{Tr}(\mathbf{A}^N)$ [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 A.

  Z is approximated in terms of the principal eigenvalue of 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.



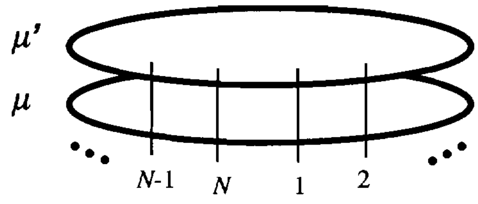
$$\mu = \{\mu_1, \mu_2, \ldots, \mu_{N-1}, \mu_N\},\ \mu_{N+1} = \mu_1.$$
$$\mu_i \in \{1, 2, \ldots, L\}, \forall i.$$
$$X \equiv \mu.$$
$$\rho(\mu_i, \mu_{i+1}) = \begin{cases} 1 & \text{if } \mu_i = \mu_{i+1}, \\ -1 & \text{otherwise.} \end{cases}$$
$$Q(\mu_i, \mu_{i+1}) = e^{q\rho(\mu_i, \mu_{i+1})}.$$
$$E(X) = -\sum_{i=1}^{N} \rho(\mu_i, \mu_{i+1}).$$
$$\begin{aligned} Z_{1,N} &= \sum_{\{X\}} \exp(-E(X)) \\ &= \sum_{\{X\}} \prod_{i=1}^{N} Q(\mu_i, \mu_{i+1}). \end{aligned}$$
$$\mathbf{A} = \begin{bmatrix} e^{q} & e^{-q} & \cdots & e^{-q} \\ e^{-q} & e^{q} & \cdots & e^{-q} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-q} & e^{-q} & \cdots & e^{q} \end{bmatrix}_{L\times L}.$$
$$l_1 = (\exp(2q) + L - 1)\exp(-q),$$
$$l_2 = l_3 = \ldots = l_L = \exp(q) - \exp(-q).$$

Fig. **3.4:** Solution of the 1×N lattice model for external field h=O – transfer matrix **A,** eigenvalues l_i.

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

$$
\begin{aligned}
C_{1,N} &= \frac{1}{NZ_{1,N}}\frac{dZ_{1,N}}{dq} \\
&= -1 + \frac{2\exp(2q)}{\exp(2q)+L-1}\left[\frac{1+(L-1)\left(\dfrac{\exp(2q)-1}{\exp(2q)+L-1}\right)^{N-1}}{1+(L-1)\left(\dfrac{\exp(2q)-1}{\exp(2q)+L-1}\right)^{N}}\right] \\
&\approx -1 + \frac{2\exp(2q)}{\exp(2q)+L-1}.
\end{aligned}
$$

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



$$\mu = \{\mu_1, \mu_2, \dots, \mu_{N-1}, \mu_N\},\ \mu_{N+1} = \mu_1.$$

$$\mu' = \{\mu_1', \mu_2', \dots, \mu_{N-1}', \mu_N'\},\ \mu_{N+1}' = \mu_1'.$$

$$\mu_i, \mu_i' \in \{-1,1\}, \forall i.$$

$$X \equiv \overbrace{\mu\,\mu'}.$$

$$E(X) = -q\sum_{i=1}^{N}\left(\mu_i\mu_{i+1} + \mu_i'\,\mu_{i+1}' + \mu_i\mu_i'\right).$$

$$Q(\mu_i, \mu_i', \mu_{i+1}, \mu_{i+1}') = \exp\left[q\left(\mu_i\mu_{i+1} + \mu_i'\,\mu_{i+1}' + \frac{1}{2}\mu_i\mu_i' + \frac{1}{2}\mu_{i+1}\mu_{i+1}'\right)\right].$$

$$Z_{2,N} = \sum_{\{X\}}\exp(-E(X))$$

$$= \sum_{\{X\}}\prod_{i=1}^{N} Q(\mu_i, \mu_i', \mu_{i+1}, \mu_{i+1}').$$

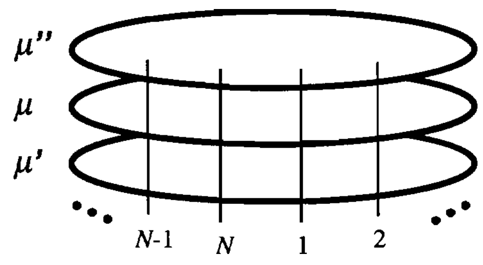
$$\mathbf{A} = \begin{bmatrix} e^{3q} & 1 & 1 & e^{-q} \\ 1 & e^{q} & e^{-3q} & 1 \\ 1 & e^{-3q} & e^{q} & 1 \\ e^{-q} & 1 & 1 & e^{3q} \end{bmatrix}.$$

$$a = e^{q}.$$

Fig. 3.5: Solution of the 2×N lattice model for external field h=O – transfer matrix **A.**

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

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



$$\mu = \{\mu_1, \mu_2, \ldots, \mu_{N-1}, \mu_N\},\ \mu_{N+1} = \mu_1.$$
$$\mu' = \{\mu_1', \mu_2', \ldots, \mu_{N-1}', \mu_N'\},\ \mu_{N+1}' = \mu_1'.$$
$$\mu'' = \{\mu_1'', \mu_2'', \ldots, \mu_{N-1}'', \mu_N''\},\ \mu_{N+1}'' = \mu_1''.$$
$$\mu_i, \mu_i', \mu_i'' \in \{-1, 1\},\ \forall i.$$

$$X \equiv \overbrace{\mu''\,\mu\,\mu'}.$$

$$E(X) = -q\sum_{i=1}^{N}\left(\mu_i\mu_i' + \mu_i\mu_i'' + \mu_i\mu_{i+1} + \mu_i'\mu_{i+1}' + \mu_i''\mu_{i+1}''\right).$$

$$Q\begin{pmatrix}\mu_i, \mu_i', \mu_i'', \\ \mu_{i+1}, \mu_{i+1}', \mu_{i+1}''\end{pmatrix} = \exp\left[q\begin{pmatrix}\mu_i\mu_{i+1} + \mu_i'\mu_{i+1}' + \mu_i''\mu_{i+1}'' \\ +\frac{1}{2}\mu_i\mu_i' + \frac{1}{2}\mu_i\mu_i'' + \frac{1}{2}\mu_{i+1}\mu_{i+1}' + \frac{1}{2}\mu_{i+1}\mu_{i+1}''\end{pmatrix}\right].$$

$$Z_{2,N} = \sum_{\{X\}}\exp(-E(X))$$
$$= \sum_{\{X\}}\prod_{i=1}^{N} Q(\mu_i, \mu_i', \mu_i'', \mu_{i+1}, \mu_{i+1}', \mu_{i+1}'').$$

$$\mathbf{A} = \begin{bmatrix} e^{5q} & e^{2q} & e^{2q} & e^{q} & e^{-q} & 1 & 1 & e^{-q} \\ e^{2q} & e^{3q} & e^{-q} & e^{-2q} & 1 & e^{q} & e^{-3q} & 1 \\ e^{2q} & e^{-q} & e^{3q} & e^{-2q} & 1 & e^{-3q} & e^{q} & 1 \\ e^{q} & e^{-2q} & e^{-2q} & e^{q} & e^{-5q} & 1 & 1 & e^{-q} \\ e^{-q} & 1 & 1 & e^{-5q} & e^{q} & e^{-2q} & e^{-2q} & e^{q} \\ 1 & e^{q} & e^{-3q} & 1 & e^{-2q} & e^{3q} & e^{-q} & e^{2q} \\ 1 & e^{-3q} & e^{q} & 1 & e^{-2q} & e^{-q} & e^{3q} & e^{2q} \\ e^{-q} & 1 & 1 & e^{-q} & e^{q} & e^{2q} & e^{2q} & e^{5q} \end{bmatrix}.$$

Fig. **3.6:** Solution of the $3\times N$ lattice model for external field $\boldsymbol{h=0}$ – transfer matrix **A.**

$$a = e^{q},$$

$$m_1 = \frac{a^4 - 1}{3a^5},$$

$$m_2 = \frac{a^2 + 1}{3a^5},$$

$$p_1 = -12a^{16} + 12a^{14} + 80a^{12} - 120a^{10} - 120a^8 + 80a^6 + 12a^4 - 12a^2 \\ + 8 + 8a^{18},$$

$$q_1 = 12a^2 \begin{pmatrix} 3 - 18a^2 + 51a^4 + 12a^8 - 60a^6 + 126a^{12} + 84a^{10} + 126a^{16} \\ +12a^{20} - 396a^{14} + 51a^{24} + 3a^{28} - 18a^{26} - 60a^{22} + 84a^{18} \end{pmatrix}^{0.5},$$

$$p_2 = \begin{pmatrix} 8 - 36a^{22} + 72a^{18} - 36a^{14} - 36a^{10} + 72a^6 - 36a^2 - 96a^{16} \\ +592a^{12} - 96a^8 + 8a^{24} + 48a^{20} + 48a^4 \end{pmatrix},$$

$$q_2 = 12a^2 \begin{pmatrix} 3a^{40} - 24a^2 + 3 + 84a^4 + 6a^8 - 132a^6 - 306a^{12} + 336a^{10} + 1287a^{16} \\ -2148a^{20} - 744a^{14} + 1287a^{24} - 306a^{28} + 84a^{36} + 6a^{32} + 336a^{30} \\ -132a^{34} - 24a^{38} - 744a^{26} + 564a^{22} + 564a^{18} \end{pmatrix}^{0.5}.$$

For $j = 1,2$

$$\theta_j = \frac{1}{3}\tan^{-1}\frac{q_j}{p_j} + \cos^{-1}\operatorname{sign}(p_j),$$

$$x_j = \left(p_j^{\,2} + q_j^{\,2}\right)^{1/6}\cos\theta_j,$$

$$y_j = \left(p_j^{\,2} + q_j^{\,2}\right)^{1/6}\sin\theta_j.$$

$$l_1 = \frac{1 - a^2 - a^4 + a^6}{a^3},$$

$$l_2 = \frac{1 - a^2 + a^4 + a^6}{a^3},$$

$$l_3 = m_1\left(x_1 + 1 + a^2 + a^4 + a^6\right),$$

$$l_7 = m_1\left(-\frac{x_1}{2} + 1 + a^2 + a^4 + a^6 + \frac{\sqrt{3}y_1}{2}\right),$$

$$l_8 = m_1\left(-\frac{x_1}{2} + 1 + a^2 + a^4 + a^6 - \frac{\sqrt{3}y_1}{2}\right),$$

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

$$l_4 = m_2\left(x_2 + 1 + 2a^4 + a^8\right),$$

$$l_5 = m_2\left(-\frac{x_2}{2} + 1 + 2a^4 + a^8 + \frac{\sqrt{3}y_2}{2}\right),$$

$$l_6 = m_2\left(-\frac{x_2}{2} + 1 + 2a^4 + a^8 - \frac{\sqrt{3}y_2}{2}\right).$$

$$Z_{3,N} = \sum_{j=1}^{8} l_j^{\,N} \approx l_4^{\,N}.$$

$$\frac{dm_2}{da} = \frac{1}{3}\left(-3a^{-4} - 5a^{-6}\right).$$

$$\frac{dp_2}{da} = \begin{pmatrix} -792a^{21} + 1296a^{17} - 504a^{13} - 360a^{9} + 432a^{5} - 72a - 1536a^{15} + 7104a^{11} \\ -768a^{7} + 192a^{23} + 960a^{19} + 192a^{3} \end{pmatrix}.$$

$$\frac{dq_2}{da} = \frac{2q_2}{a} + \left(\frac{3}{q_2}\right)^{0.5} \begin{pmatrix} 120a^{40} - 48a^{2} + 336a^{4} + 48a^{8} - 792a^{6} - 3672a^{12} + 3360a^{10} \\ +20592a^{16} - 42960a^{20} - 10416a^{14} + 30888a^{24} - 8568a^{28} \\ +3024a^{36} + 192a^{32} + 10080a^{30} - 4488a^{34} - 912a^{38} \\ -19344a^{26} + 12408a^{22} + 10152a^{18} \end{pmatrix}.$$

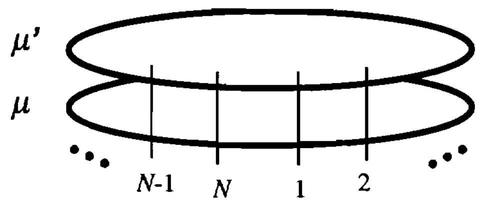
$$\frac{dx_2}{da} = \frac{1}{6}\left(p_2^{\,2} + q_2^{\,2}\right)^{-5/6}\left[2p_2\frac{dp_2}{da} + 2q_2\frac{dq_2}{da}\right]\operatorname{sign}(p_2)\left|\cos\theta_2\right|$$

$$-\frac{1}{3}\left(p_2^{\,2} + q_2^{\,2}\right)^{1/6}\sin\theta_2 \frac{p_2\dfrac{dq_2}{da} - q_2\dfrac{dp_2}{da}}{p_2^{\,2} + q_2^{\,2}}$$

$$\frac{dl_4}{da} = \frac{dm_2}{da}\left(x_2 + 1 + 2a^4 + a^8\right) + m_2\left(\frac{dx_2}{da} + 8a^3 + 8a^7\right)$$

$$C_{3,N} = \frac{1}{NZ_{3,N}}\frac{dZ_{3,N}}{dq} \approx \frac{a}{l_4}\frac{dl_4}{da}.$$

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



$$\mu = \{\mu_1, \mu_2, \ldots, \mu_{N-1}, \mu_N\},\ \mu_{N+1} = \mu_1.$$

$$\mu' = \{\mu_1', \mu_2', \ldots, \mu_{N-1}', \mu_N'\},\ \mu_{N+1}' = \mu_1'.$$

$$\mu_i, \mu_i' \in \{-1, 1\}, \forall i.$$

$$X \equiv \widehat{\mu' \ \mu}.$$

$$E(X) = -q\sum_{i=1}^{N}(\mu_i\mu_{i+1} + \mu_i'\mu_{i+1}' + \mu_i\mu_i') - h\sum_{i=1}^{N}(\mu_i + \mu_i').$$

$$Q(\mu_i, \mu_i', \mu_{i+1}, \mu_{i+1}') = \exp\left[q\left(\mu_i\mu_{i+1} + \mu_i'\mu_{i+1}' + \frac{1}{2}\mu_i\mu_i' + \frac{1}{2}\mu_{i+1}\mu_{i+1}'\right) + \frac{h}{2}(\mu_i + \mu_{i+1} + \mu_i' + \mu_{i+1}')\right].$$

$$Z_{2,N} = \sum_{\{X\}}\exp(-E(X))$$

$$= \sum_{\{X\}}\prod_{i=1}^{N} Q(\mu_i, \mu_i', \mu_{i+1}, \mu_{i+1}').$$

$$\mathbf{A} = \begin{bmatrix} e^{3q+2h} & e^{h} & e^{h} & e^{-q} \\ e^{h} & e^{q} & e^{-3q} & e^{-h} \\ e^{h} & e^{-3q} & e^{q} & e^{-h} \\ e^{-q} & e^{-h} & e^{-h} & e^{3q-2h} \end{bmatrix}$$

$$x_2 = -2e^{3q}\cosh 2h - e^{q} - e^{-3q},$$

$$A = 6(e^{4q} - 5)\cosh 2h + 6\sinh 6q - 6\cosh 2q - 6e^{6q}\cosh 4h - 12e^{-2q},$$

$$B = -\begin{bmatrix} 18(e^{9q} + e^{5q} - e^{q} - e^{-3q})\cosh 2h + 18(e^{7q} - e^{3q})\cosh 4h + 18e^{-5q} \\ +72e^{3q} - 90e^{-q} + 2x_2^{\,3} \end{bmatrix},$$

$$t = \left[\frac{-9B + (81B^2 + 12A^3)^{0.5}}{18}\right]^{1/3}.$$

Fig. 3.7: General solution of the 2×*N* lattice model – transfer matrix A.

$$\bar{l}_2 = \frac{A}{3t} - t,$$

$$\bar{l}_3 = -\frac{\bar{l}_2}{2} + \frac{1}{2}\left(-4A - 3\bar{l}_2^{\,2}\right)^{0.5},$$

$$\bar{l}_4 = -\frac{\bar{l}_2}{2} - \frac{1}{2}\left(-4A - 3\bar{l}_2^{\,2}\right)^{0.5}.$$

$$l_1 = e^{q} - e^{-3q},$$

$$l_2 = \frac{\bar{l}_2 - x_2}{3},$$

$$l_3 = \frac{\bar{l}_3 - x_2}{3},$$

$$l_4 = \frac{\bar{l}_4 - x_2}{3}.$$

$$\frac{dl_3}{ds} = \frac{1}{3}\frac{d\bar{l}_3}{ds} - \frac{1}{3}\frac{dx_2}{ds},$$

$$\frac{d\bar{l}_3}{ds} = -\frac{1}{2}\frac{d\bar{l}_2}{ds} + \frac{1}{4}\left(-4A - 3\bar{l}_2^{\,2}\right)^{-0.5}\left(-4\frac{dA}{ds} - 6\bar{l}_2\frac{d\bar{l}_2}{ds}\right),$$

$$\frac{d\bar{l}_2}{ds} = \frac{1}{3t}\frac{dA}{ds} - \frac{dt}{ds}\left(\frac{A}{3t^2} + 1\right),$$

$$\frac{dt}{ds} = \frac{1}{54t^2}\left[-9\frac{dB}{ds} + \frac{81B\frac{dB}{ds} + 18A^2\frac{dA}{ds}}{\left(81B^2 + 12A^3\right)^{0.5}}\right].$$

$$\frac{dA}{dh} = 12\left(e^{4q} - 5\right)\sinh 2h - 24e^{6q}\sinh 4h,$$

$$\frac{dA}{dq} = 24e^{4q}\cosh 2h + 36\cosh 6q - 12\sinh 2q - 36e^{6q}\cosh 4h + 24e^{-2q}.$$

$$\frac{dx_2}{dh} = -4e^{3q}\sinh 2h,$$

$$\frac{dx_2}{dq} = -6e^{3q}\cosh 2h - e^{q} + 3e^{-3q}.$$

Fig. **3.7** (contd.): General solution of the $2\times N$ lattice model – principal eigenvalue l_1.

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

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

$$\begin{aligned} Z_{2,N} &= \mathrm{Tr}\left(\mathbf{A}^N\right) \\ &= \sum_{i=1}^{4} {l_i}^N \approx {l_3}^N \end{aligned}$$

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

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

Fig. **3.7** (contd.): General solution of the 2×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].

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

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

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	0.9694	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 3.2 (contd.)
Average correlation per pixel, $C_{2,N}$, 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	0.4412	0.4778	0.5133	0.5475	0.5803	0.6114	0.6410	0.6689
0.0500	0.5292	0.5641	0.5973	0.6287	0.6582	0.6860	0.7119	0.7360
0.1000	0.6123	0.6441	0.6738	0.7016	0.7272	0.7510	0.7729	0.7931
0.1500	0.6877	0.7156	0.7412	0.7647	0.7862	0.8059	0.8238	0.8402
0.2000	0.7537	0.7771	0.7983	0.8176	0.8350	0.8508	0.8651	0.8780
0.2500	0.8091	0.8281	0.8452	0.8605	0.8743	0.8866	0.8978	0.9078
0.3000	0.8543	0.8693	0.8826	0.8945	0.9051	0.9146	0.9232	0.9308
0.3500	0.8901	0.9016	0.9118	0.9209	0.9289	0.9361	0.9426	0.9483
0.4000	0.9178	0.9265	0.9342	0.9410	0.9471	0.9525	0.9573	0.9616
0.4500	0.9389	0.9454	0.9511	0.9562	0.9607	0.9647	0.9683	0.9715
0.5000	0.9548	0.9596	0.9638	0.9676	0.9709	0.9739	0.9765	0.9789
0.5500	0.9666	0.9702	0.9733	0.9761	0.9785	0.9807	0.9827	0.9844
0.6000	0.9754	0.9780	0.9803	0.9823	0.9841	0.9858	0.9872	0.9885
0.6500	0.9819	0.9838	0.9855	0.9870	0.9883	0.9895	0.9905	0.9915
0.7000	0.9867	0.9881	0.9893	0.9904	0.9914	0.9922	0.9930	0.9937
0.7500	0.9902	0.9912	0.9921	0.9929	0.9936	0.9943	0.9948	0.9954
0.8000	0.9928	0.9935	0.9942	0.9948	0.9953	0.9958	0.9962	0.9966
0.8500	0.9947	0.9952	0.9957	0.9962	0.9965	0.9969	0.9972	0.9975
0.9000	0.9961	0.9965	0.9968	0.9972	0.9974	0.9977	0.9979	0.9981
0.9500	0.9971	0.9974	0.9977	0.9979	0.9981	0.9983	0.9985	0.9986
1.0000	0.9979	0.9981	0.9983	0.9985	0.9986	0.9987	0.9989	0.9990
1.0500	0.9984	0.9986	0.9987	0.9989	0.9990	0.9991	0.9992	0.9992
1.1000	0.9989	0.9990	0.9991	0.9992	0.9992	0.9993	0.9994	0.9994
1.1500	0.9992	0.9992	0.9993	0.9994	0.9994	0.9995	0.9995	0.9996
1.2000	0.9994	0.9994	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997
1.2500	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997	0.9998	0.9998
1.3000	0.9997	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998
1.3500	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999
1.4000	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999
1.4500	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999

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

q⇓ h⇒	1.2000	1.2500	1.3000	1.3500	1.4000	1.4500	1.5000	1.5500
0.0001	0.6952	0.7198	0.7427	0.7641	0.7840	0.8024	0.8194	0.8351
0.0500	0.7584	0.7792	0.7983	0.8160	0.8323	0.8472	0.8609	0.8734
0.1000	0.8116	0.8286	0.8441	0.8583	0.8713	0.8831	0.8939	0.9037
0.1500	0.8551	0.8686	0.8809	0.8921	0.9022	0.9114	0.9198	0.9273
0.2000	0.8897	0.9003	0.9098	0.9185	0.9263	0.9333	0.9397	0.9455
0.2500	0.9168	0.9249	0.9322	0.9388	0.9447	0.9501	0.9549	0.9592
0.3000	0.9376	0.9438	0.9493	0.9543	0.9587	0.9628	0.9664	0.9696
0.3500	0.9535	0.9581	0.9622	0.9660	0.9693	0.9723	0.9750	0.9774
0.4000	0.9654	0.9689	0.9719	0.9747	0.9772	0.9794	0.9815	0.9833
0.4500	0.9744	0.9769	0.9792	0.9813	0.9831	0.9848	0.9863	0.9876
0.5000	0.9810	0.9829	0.9846	0.9861	0.9875	0.9887	0.9898	0.9908
0.5500	0.9860	0.9874	0.9886	0.9897	0.9907	0.9916	0.9925	0.9932
0.6000	0.9896	0.9907	0.9916	0.9924	0.9932	0.9938	0.9944	0.9950
0.6500	0.9923	0.9931	0.9938	0.9944	0.9949	0.9954	0.9959	0.9963
0.7000	0.9943	0.9949	0.9954	0.9959	0.9963	0.9966	0.9969	0.9972
0.7500	0.9958	0.9962	0.9966	0.9969	0.9972	0.9975	0.9977	0.9980
0.8000	0.9969	0.9972	0.9975	0.9977	0.9980	0.9982	0.9983	0.9985
0.8500	0.9977	0.9979	0.9981	0.9983	0.9985	0.9986	0.9988	0.9989
0.9000	0.9983	0.9985	0.9986	0.9988	0.9989	0.9990	0.9991	0.9992
0.9500	0.9988	0.9989	0.9990	0.9991	0.9992	0.9993	0.9993	0.9994
1.0000	0.9991	0.9992	0.9992	0.9993	0.9994	0.9994	0.9995	0.9995
1.0500	0.9993	0.9994	0.9994	0.9995	0.9995	0.9996	0.9996	0.9997
1.1000	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997	0.9998
1.1500	0.9996	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998
1.2000	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999
1.2500	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999
1.3000	0.9998	0.9999	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	1.0000	1.0000	1.0000
1.4500	0.9999	0.9999	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

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

$q\Downarrow$ $h\Rightarrow$	1.6000	1.6500	1.7000	1.7500	1.8000	1.8500	1.9000	1.9500
0.0001	0.8496	0.8629	0.8751	0.8863	0.8965	0.9059	0.9145	0.9223
0.0500	0.8849	0.8954	0.9049	0.9137	0.9216	0.9289	0.9355	0.9415
0.1000	0.9127	0.9208	0.9282	0.9349	0.9410	0.9465	0.9515	0.9561
0.1500	0.9342	0.9404	0.9461	0.9512	0.9558	0.9600	0.9638	0.9672
0.2000	0.9507	0.9554	0.9597	0.9635	0.9670	0.9701	0.9730	0.9756
0.2500	0.9632	0.9667	0.9699	0.9728	0.9754	0.9778	0.9799	0.9818
0.3000	0.9726	0.9752	0.9776	0.9798	0.9817	0.9835	0.9851	0.9865
0.3500	0.9796	0.9816	0.9834	0.9850	0.9864	0.9877	0.9889	0.9900
0.4000	0.9849	0.9864	0.9877	0.9889	0.9899	0.9909	0.9918	0.9926
0.4500	0.9888	0.9899	0.9909	0.9918	0.9925	0.9933	0.9939	0.9945
0.5000	0.9917	0.9925	0.9932	0.9939	0.9945	0.9950	0.9955	0.9959
0.5500	0.9939	0.9945	0.9950	0.9955	0.9959	0.9963	0.9967	0.9970
0.6000	0.9955	0.9959	0.9963	0.9967	0.9970	0.9973	0.9975	0.9978
0.6500	0.9966	0.9970	0.9973	0.9975	0.9978	0.9980	0.9982	0.9983
0.7000	0.9975	0.9978	0.9980	0.9982	0.9983	0.9985	0.9986	0.9988
0.7500	0.9982	0.9983	0.9985	0.9986	0.9988	0.9989	0.9990	0.9991
0.8000	0.9986	0.9988	0.9989	0.9990	0.9991	0.9992	0.9993	0.9993
0.8500	0.9990	0.9991	0.9992	0.9993	0.9993	0.9994	0.9995	0.9995
0.9000	0.9993	0.9993	0.9994	0.9994	0.9995	0.9996	0.9996	0.9996
0.9500	0.9994	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997
1.0000	0.9996	0.9996	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998
1.0500	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998	0.9999
1.1000	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999
1.1500	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.2000	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.2500	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	1.0000	1.0000
1.3000	0.9999	0.9999	0.9999	1.0000	1.0000	1.0000	1.0000	1.0000
1.3500	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.4000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.4500	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

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

q⇓ h⇒	0.0001	0.0500	0.1000	0.1500	0.2000	0.2500	0.3000	0.3500
0.00	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.[illegible]419	0.7956	0.7976	0.8400	0.8703
0.5000	0.0007	0.3464	0.5844	0.7229	0.8023	0.8512	0.8833	0.9057
0.5500	0.0010	0.4263	0.6731	0.793[illegible]	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.[illegible]760	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	0.99[illegible]1	0. 968	0.9973	0. 9
1.1000	0.0130	0.9857	0.9945	0.9964	0.[illegible]72	0.9977	0.9981	0. [illegible]
1.1500	0.0163	0.9905	0.9962	0.9975	0.99[illegible]0	0.9984	0.9986	0. 9
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	0.99[illegible]7	0.9991	0.9993	0.9994	0 0.99	0.9995
1.3500	0.0387	0.9981	0.[illegible]91	0.9994	0.[illegible]995	0.9996	0 0.99	0.9997
1.4000	0.0479	0.9987	0.9[illegible]94	0.9995	0.9996	0.9997	0 0.99	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

$q \Downarrow$ $h \Rightarrow$	0.4000	0.4500	0.5000	0.5500	0.6000	0.6500	0.7000	0.7500
0.0001	0.3801	0.4221	0.4623	0.5007	0.5372	0.5719	0.6045	0.6353
0.0500	0.4321	0.4770	0.5192	0.5589	0.5959	0.6303	0.6622	0.6918
0.1000	0.4901	0.5370	0.5802	0.6198	0.6559	0.6889	0.7188	0.7459
0.1500	0.5535	0.6008	0.6432	0.6811	0.7149	0.7450	0.7719	0.7957
0.2000	0.6201	0.6657	0.7055	0.7401	0.7702	0.7965	0.8194	0.8396
0.2500	0.6869	0.7287	0.7640	0.7940	0.8195	0.8414	0.8602	0.8765
0.3000	0.7503	0.7864	0.8161	0.8408	0.8615	0.8790	0.8938	0.9065
0.3500	0.8069	0.8364	0.8602	0.8797	0.8957	0.9092	0.9205	0.9301
0.4000	0.8545	0.8775	0.8958	0.9105	0.9226	0.9327	0.9411	0.9483
0.4500	0.8927	0.9099	0.9234	0.9343	0.9431	0.9505	0.9567	0.9620
0.5000	0.9221	0.9345	0.9443	0.9521	0.9585	0.9638	0.9683	0.9721
0.5500	0.9439	0.9528	0.9597	0.9653	0.9698	0.9737	0.9769	0.9796
0.6000	0.9599	0.9661	0.9709	0.9749	0.9781	0.9809	0.9832	0.9851
0.6500	0.9714	0.9757	0.9791	0.9818	0.9842	0.9861	0.9877	0.9892
0.7000	0.9796	0.9825	0.9849	0.9869	0.9885	0.9899	0.9911	0.9921
0.7500	0.9854	0.9875	0.9891	0.9905	0.9917	0.9926	0.9935	0.9942
0.8000	0.9896	0.9910	0.9922	0.9931	0.9939	0.9946	0.9953	0.9958
0.8500	0.9925	0.9935	0.9943	0.9950	0.9956	0.9961	0.9965	0.9969
0.9000	0.9946	0.9953	0.9959	0.9964	0.9968	0.9972	0.9975	0.9977
0.9500	0.9961	0.9966	0.9970	0.9974	0.9977	0.9979	0.9981	0.9983
1.0000	0.9972	0.9975	0.9978	0.9981	0.9983	0.9985	0.9986	0.9988
1.0500	0.9980	0.9982	0.9984	0.9986	0.9988	0.9989	0.9990	0.9991
1.1000	0.9985	0.9987	0.9988	0.9990	0.9991	0.9992	0.9993	0.9993
1.1500	0.9989	0.9991	0.9992	0.9992	0.9993	0.9994	0.9995	0.9995
1.2000	0.9992	0.9993	0.9994	0.9995	0.9995	0.9996	0.9996	0.9996
1.2500	0.9994	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997
1.3000	0.9996	0.9996	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998
1.3500	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998	0.9999
1.4000	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999
1.4500	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999

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

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

$q\Downarrow$ $h\Rightarrow$	1.2000	1.2500	1.3000	1.3500	1.4000	1.4500	1.5000	1.5500
0.0001	0.8338	0.8484	0.8618	0.8741	0.8854	0.8958	0.9052	0.9138
0.0500	0.8690	0.8812	0.8922	0.9023	0.9114	0.9197	0.9273	0.9341
0.1000	0.8985	0.9083	0.9171	0.9251	0.9323	0.9389	0.9447	0.9500
0.1500	0.9224	0.9301	0.9370	0.9432	0.9488	0.9538	0.9584	0.9624
0.2000	0.9413	0.9472	0.9525	0.9573	0.9616	0.9654	0.9688	0.9719
0.2500	0.9559	0.9605	0.9645	0.9681	0.9713	0.9742	0.9767	0.9790
0.3000	0.9671	0.9705	0.9736	0.9762	0.9786	0.9808	0.9827	0.9844
0.3500	0.9756	0.9781	0.9804	0.9824	0.9842	0.9858	0.9872	0.9885
0.4000	0.9820	0.9838	0.9855	0.9870	0.9883	0.9895	0.9905	0.9915
0.4500	0.9867	0.9880	0.9893	0.9904	0.9913	0.9922	0.9930	0.9937
0.5000	0.9902	0.9912	0.9921	0.9929	0.9936	0.9942	0.9948	0.9953
0.5500	0.9928	0.9935	0.9942	0.9947	0.9953	0.9957	0.9962	0.9965
0.6000	0.9947	0.9952	0.9957	0.9961	0.9965	0.9969	0.9972	0.9974
0.6500	0.9961	0.9965	0.9968	0.9971	0.9974	0.9977	0.9979	0.9981
0.7000	0.9971	0.9974	0.9977	0.9979	0.9981	0.9983	0.9985	0.9986
0.7500	0.9979	0.9981	0.9983	0.9984	0.9986	0.9987	0.9989	0.9990
0.8000	0.9984	0.9986	0.9987	0.9989	0.9990	0.9991	0.9992	0.9992
0.8500	0.9988	0.9990	0.9991	0.9992	0.9992	0.9993	0.9994	0.9994
0.9000	0.9991	0.9992	0.9993	0.9994	0.9994	0.9995	0.9995	0.9996
0.9500	0.9994	0.9994	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997
1.0000	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997	0.9997	0.9998
1.0500	0.9997	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998
1.1000	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999
1.1500	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999
1.2000	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.2500	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.3000	0.9999	0.9999	0.9999	0.9999	0.9999	1.0000	1.0000	1.0000
1.3500	0.9999	0.9999	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.4000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.4500	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000

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

$q\Downarrow$ $h\Rightarrow$	1.6000	1.6500	1.7000	1.7500	1.8000	1.8500	1.9000	1.9500
0.0001	0.9217	0.9289	0.9355	0.9414	0.9468	0.9518	0.9563	0.9603
0.0500	0.9403	0.9459	0.9510	0.9556	0.9598	0.9636	0.9671	0.9702
0.1000	0.9548	0.9592	0.9631	0.9666	0.9698	0.9727	0.9753	0.9777
0.1500	0.9661	0.9694	0.9723	0.9750	0.9774	0.9796	0.9816	0.9834
0.2000	0.9746	0.9771	0.9794	0.9814	0.9832	0.9848	0.9863	0.9876
0.2500	0.9811	0.9830	0.9 48	0.9862	0.9875	0.9887	0.9898	0.9908
0.3000	0.9860	0.9874	0.9886	0.9897	0.9907	0.9916	0.9924	0.9932
0.3500	0.9896	0.9906	0.9916	0.9924	0.9931	0.9938	0.9944	0.9949
0.4000	0.9923	0.9931	0.9937	0.9944	0.9949	0.9954	0.9959	0.9963
0.4500	0.9943	0.9949	0.9954	0.9958	0.9962	0.9966	0.9969	0.9972
0.5000	0.9958	0.9962	0.9966	0.9969	0.9972	0.9975	0.9977	0.9979
0.5500	0.9969	0.9972	0.9975	0.9977	0.9 79	0.9981	0.9983	0.9985
0.6000	0.9977	0.9979	0.9981	0.9983	0.9985	0.9986	0.9988	0.9989
0.6500	0.9983	0.9985	0.9986	0.9988	0.9989	0.9990	0.9991	0.9992
0.7000	0.9987	0.9989	0.9990	0.9991	0.9992	0.9992	0.9993	0.9994
0.7500	0.9991	0.9992	0.9992	0.9993	0.9994	0.9994	0.9995	0.9995
0.8000	0.9993	0.9994	0.9994	0.9995	0.9 95	0.9996	0.9996	0.9997
0.8500	0.9995	0.9995	0.9996	0.9996	0.9997	0.9997	0.9997	0.9998
0.9000	0.9996	0.9997	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998
0.9500	0.9997	0.9997	0.9998	0.9998	0.9998	0.9998	0.9998	0.9999
1.0000	0.9998	0.9998	0.9998	0.9998	0.9999	0.9999	0.9999	0.9999
1.0500	0.9998	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.1000	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999
1.1500	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	1.0000	1.0000
1.2000	0.9999	0.9999	0.9999	1.0000	1.0000	1.0000	1.0000	1.0000
1.2500	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.3000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
1.3500	1.0000	1 0000	1.0000	1.0000	1.0000	1 0000	1.0000	1.0000
1.4000	1.0000	1 0000	1.0000	1.0000	1.0000	1 0000	1.0000	1.0000
1.4500	1.0000	1 0000	1.0000	1.0000	1.0000	1 0000	1.0000	1.0000

4. MODEL IMPLEMENTATION

4.1 Preamble

Conventional analysis of remote sensing data proceeds in the following way.

The element to be classified/labeled is identified.

The multispectral data on the element is input to a decision rule.

The output of the decision rule is observed and the element is labeled accordingly.

The schema on the left in Figure 4.1 is a representation of the above process. The data for the unlabeled element, in gray, is fed to the rule whose decision identifies the pixel as 'black' or 'white'. In the schema on the right in Figure 4.1, it is assumed that analysis prior to the current stage has labeled (as 'black') two of the neighbors of the pixel under consideration (shown in gray). It may be surmised that the labeling of the neighbors influences that of the analyzed element, and that the decision rule can be biased to favor a 'black' labeling for the case under consideration. The right side schema in Figure 4.1 is an illustration of this reasoning.

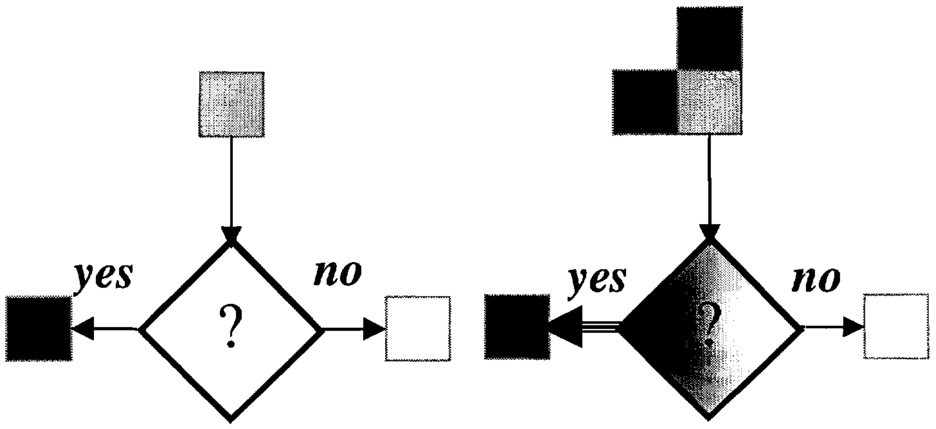


Fig. 4.1: Schemata illustrating remote sensing data classification, with and without knowledge of neighborhood.

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.

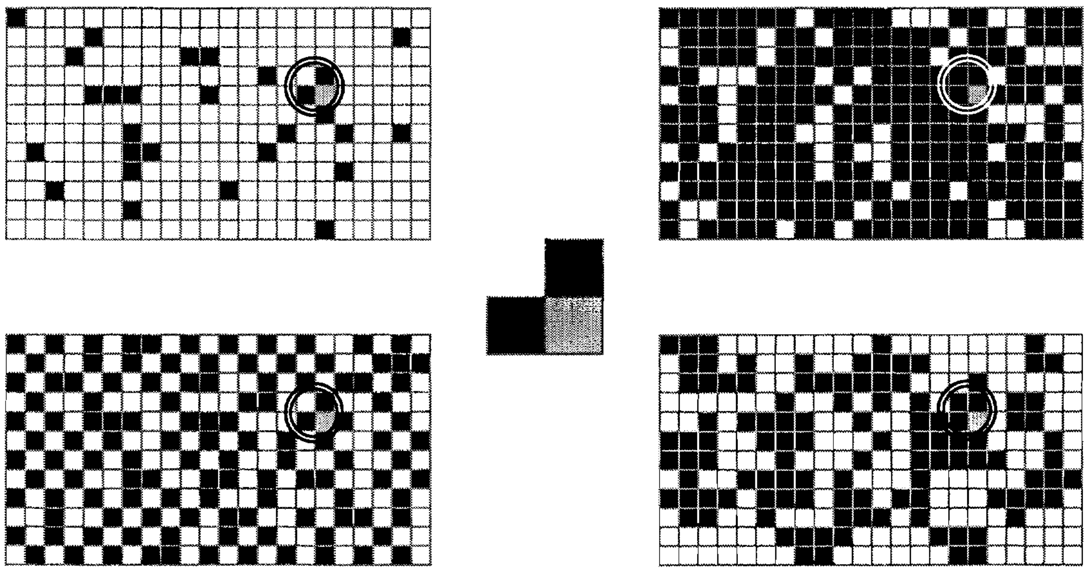


Fig. **4.2:** Central image illustrates a case where the pixel under analysis has two neighbors pre-labeled as 'black'. The four peripheral images present scenarios that could possibly yield the situation depicted in the center.

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 $h\neq0$), 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 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 property[3]. It comprises a sequence of random variables X_0, X_1, X_2, ..., 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.

$$P_X(X_t = x)\mathbf{P}(X_{t+1} = \mathrm{y} \mid X_t = \mathrm{x}) = P_X(X_t = y)\mathbf{P}(X_{t+1} = \mathrm{x} \mid X_t = y).$$

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

Let a, 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.

$$a_{xy} = \mathrm{F}\left(\frac{P_X(y)}{P_X(x)}\right), \tag{4.1}$$

where F(•) 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,

$$\frac{P_X(y)}{P_X(x)} = \exp(-\mathrm{E}(y) + \mathrm{E}(x)) = \exp(-\Delta\mathrm{E}) \tag{4.2}$$

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 simulation[4], 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 energies[5] 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.

---

[3]The Markov property cited above implies that the process is 'memoryless', i.e. $\mathbf{P}(X_{t+1}|X_t, X_{t-1},\ldots)=\mathbf{P}(X_{t+1}|X_t)$ [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.

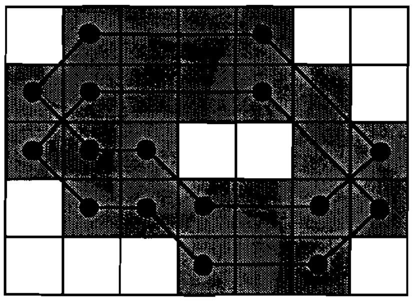


Fig. 4.3: A sample looped 2x12 chain through the given 5x7 lattice system.

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.

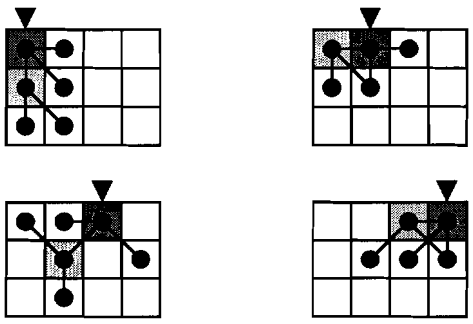
```
int chain_path(int row-count, int column-count)
{
N=min(row_count, column-count);
chain = assign_memory(1, row_count*column_count);
link_number=1;
itemp = jtemp = choose-random-number (1, row_count*column_count);
chain[link_number] = jtemp;

/*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 irnplementation 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 thr: 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)
I
/*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 X
columncount*/
set_neighbor_indices(pointer, N,E,W,S,NE,NW,SE,SW);

/*Randomly select neighbor of pointer, thus completing the chain-link*/
link-tosointer =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);
doI
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*/
doI
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-toqointer-EAST -- pointer --neighbor-to-pointer-WEST
                                      |
       neighbor-to-1ink_EAST--1ink_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.*/
```

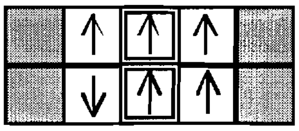
Fig. 4.6:Pseudocode for generation of a pseudo-link to a given element in a finite two-dimensional lattice.

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, AE, 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

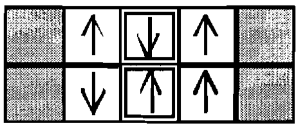


Fig. 4.7: AE = E_2-E_1 = $-q[(-3) - 3]-h[0 - 2] = 6q + 2h$.

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

$$V_i = aV_{i-1}(\text{mod } T) \quad \forall i = 1,2,3,\ldots,$$

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

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. 615,491 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				Simulated image	
	C	M	q	h	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

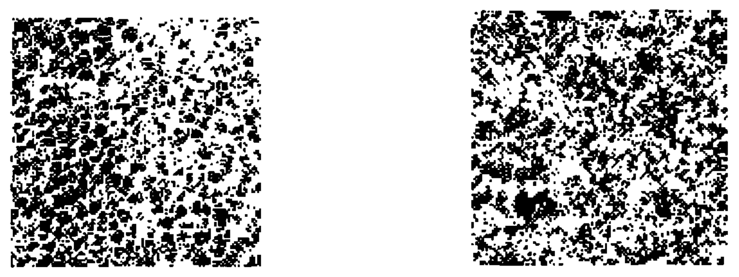


Fig. 4.8: Image 'PlasticBubs' on the left. Simulated image on the right.

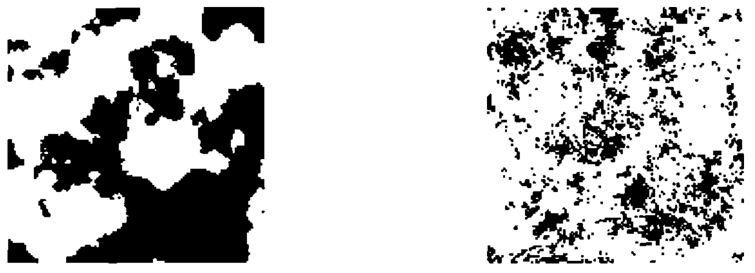


Fig. 4.9: Image 'NaturalTurbulence' on the left. Simulated image on the right.

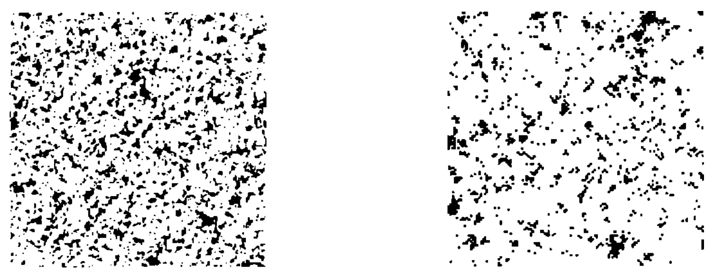


Fig. 4.10: Image 'BrodatzBeachSand' on the left. Simulated image on the right.

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 $\boldsymbol{Y}=(\boldsymbol{y}_1, y_2, \ldots, \boldsymbol{y}_N)$, with N being the total number of pixels in the image. Each y_i will denote a vector of length p (also termed the dimensionality of the data). To each y_i 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 Y will be $(x_1, \ldots, x_N)= \boldsymbol{X}\in \Lambda^N$. The ensuing discussion will incorporate the $2\times N$ chain model of Chapter 3 to model the distribution of the system state X.

A useful assumption for analysis is that y_i are multivariate normal random variables [1] [2], with distribution parameters dependent on the respective label assignment x_i. The analysis is best illustrated through the following example.

Let $\boldsymbol{X}_{\text{old}}= (x_1,\ldots, x_m=1, x_{m+1}=1,\ldots, x_N)$ and $\boldsymbol{X}_{\text{new}}= (x_1,\ldots, x_m=1, x_{m+1}=-1,\ldots, x_N)$, with the task being to determine which of the two label assignments is the more accurate labeling, given the multivariate data 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 X that is the best fit to the spectral data Y. Bayesian analysis [56] suggests that the decision should be made on the basis of the following decision rule -

$$P(\boldsymbol{X}_{\text{old}}, \boldsymbol{Y}) \underset{\boldsymbol{X}_{\text{new}}}{\overset{\boldsymbol{X}_{\text{old}}}{\gtrless}} P(\boldsymbol{X}_{\text{new}}, \boldsymbol{Y}). \tag{4.3}$$

Here, $P(\boldsymbol{X}_{\{\text{old, new}\}}, \boldsymbol{Y})$[6] represents the joint probability of occurrence of the spectral data Y, and the corresponding classification 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 $\boldsymbol{Y}|\boldsymbol{X}$ [57] (sub-script added as per requirement). The representation of the associated density functions is a:; $p(\boldsymbol{Y}|\boldsymbol{X}_{\{\text{old, new}\}})$. Simplification of expression 4.3 yields

$$P(\boldsymbol{X}_{\text{old}})p(\boldsymbol{Y} \mid \boldsymbol{X}_{\text{old}}) \underset{\boldsymbol{X}_{\text{new}}}{\overset{\boldsymbol{X}_{\text{old}}}{\gtrless}} P(\boldsymbol{X}_{\text{new}})p(\boldsymbol{Y} \mid \boldsymbol{X}_{\text{new}})$$

$$\prod_{j=1}^{N} P(\boldsymbol{X}_{\text{old}})p(\boldsymbol{y}_j \mid x_j \in \boldsymbol{X}_{\text{old}}) \underset{\boldsymbol{X}_{\text{new}}}{\overset{\boldsymbol{X}_{\text{old}}}{\gtrless}} \prod_{j=1}^{N} P(\boldsymbol{X}_{\text{new}})p(\boldsymbol{y}_j \mid x_j \in \boldsymbol{X}_{\text{new}})$$

$$P(\boldsymbol{X}_{\text{old}})p(\boldsymbol{y}_m \mid x_m = 1)p(\boldsymbol{y}_{m+1} \mid x_{m+1} = 1) \underset{\boldsymbol{X}_{\text{new}}}{\overset{\boldsymbol{X}_{\text{old}}}{\gtrless}} P(\boldsymbol{X}_{\text{new}})p(\boldsymbol{y}_m \mid x_m = 1)p(\boldsymbol{y}_{m+1} \mid x_{m+1} = -1)$$

The above simplifications are based on the following assumptions -

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

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

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.

$$-\mathrm{E}(\boldsymbol{X}_{\text{old}}) + \mathrm{E}(\boldsymbol{X}_{\text{new}}) + \ln\left[\frac{P(\boldsymbol{y}_m \mid x_m = 1)P(\boldsymbol{y}_{m+1} \mid x_{m+1} = 1)}{P(\boldsymbol{y}_m \mid x_m = 1)P(\boldsymbol{y}_{m+1} \mid x_{m+1} = -1)}\right] \underset{\boldsymbol{X}_{\text{new}}}{\overset{\boldsymbol{X}_{\text{old}}}{\gtrless}} 0,$$

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

$$\Delta\mathrm{E} + r(\boldsymbol{y}_m, \boldsymbol{y}_{m+1}) \underset{\boldsymbol{X}_{\text{new}}}{\overset{\boldsymbol{X}_{\text{old}}}{\gtrless}} 0. \tag{4.4}$$

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 {X) is enormous, and an exhaustive search for the configuration that maximizes the probability $P(\boldsymbol{X}, \mathrm{Y})$ 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, $\exp\left[-\left(\Delta E + r(y_m, y_{m+1})\right)\right]$).

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) algorithm[7] [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 Y, binary system labeling X. Y comprises data drawn from either one of two multivariate normal distributions.

STEP 2. Using the class assignment to each element in 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 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

[7]Statistical analysis of remote sensing data centers on the accurate estimation of the conditional distributions $Y|X$. 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 $\overline{\boldsymbol{Y}}$ with a multipolar labeling $\overline{\boldsymbol{X}}$, analysis may proceed in one of two ways -

> It could be surmised that one of the scene classes used in the labeling $\overline{\boldsymbol{X}}$, 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 $\overline{\boldsymbol{Y}}$ and an associated multipolar labeling $\overline{\boldsymbol{X}}$, Y c$\overline{\boldsymbol{Y}}$ and the corresponding bipolar labeling $\boldsymbol{X} \subset \overline{\boldsymbol{X}}$ 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 $\mathbf{I}_{p}$ as the $p{\times}p$ identity matrix.
- Generate 10,000 data as a sample from a multivariate Gaussian process [62] with mean q, 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 400x500 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 400x500 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.

▢ Class I

Class II

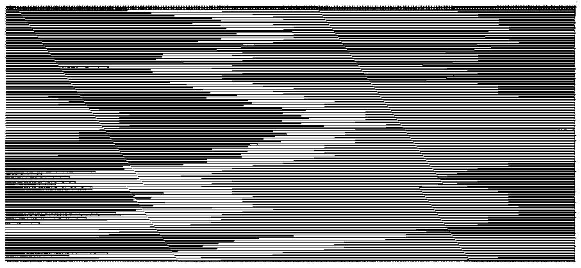


Fig. 4.11: Pictorial representation of class-label assignment for designed data.

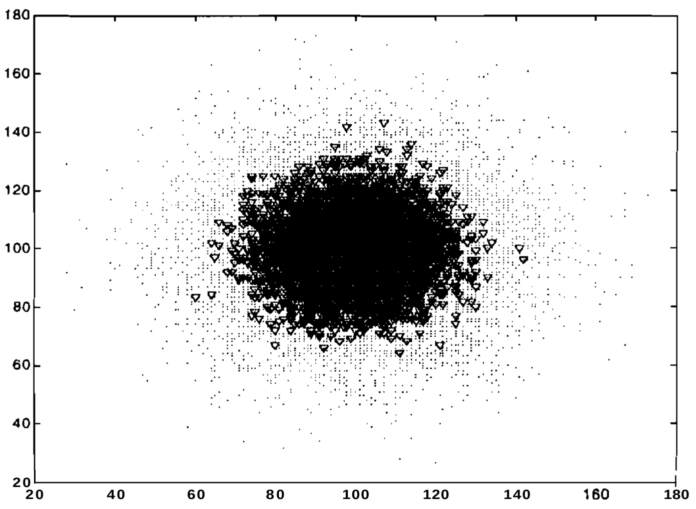


Fig. 4.12: (For p=2) Scatter plot for D_1 (as red triangles) and D_2 (as black dots).

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 –

$$r(y) = \frac{3}{800}(y-\eta)'(y-\eta) + \frac{1}{2}\ln\frac{1}{16} \overset{II}{\underset{I}{\gtrless}} 0.$$

Here, y is the p-dimensioned data on the element to be classified, and η 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

$$\begin{aligned} P_{err}^{I} &= \Pr\left(1.5\chi_p^2 + p\ln 2 > 0\right), \\ P_{err}^{II} &= \Pr\left(0.375\chi_p^2 + p\ln 2 < 0\right). \end{aligned} \tag{4.5}$$

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

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, that were wrongly identified in the mixture of D, 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 selection!; 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	

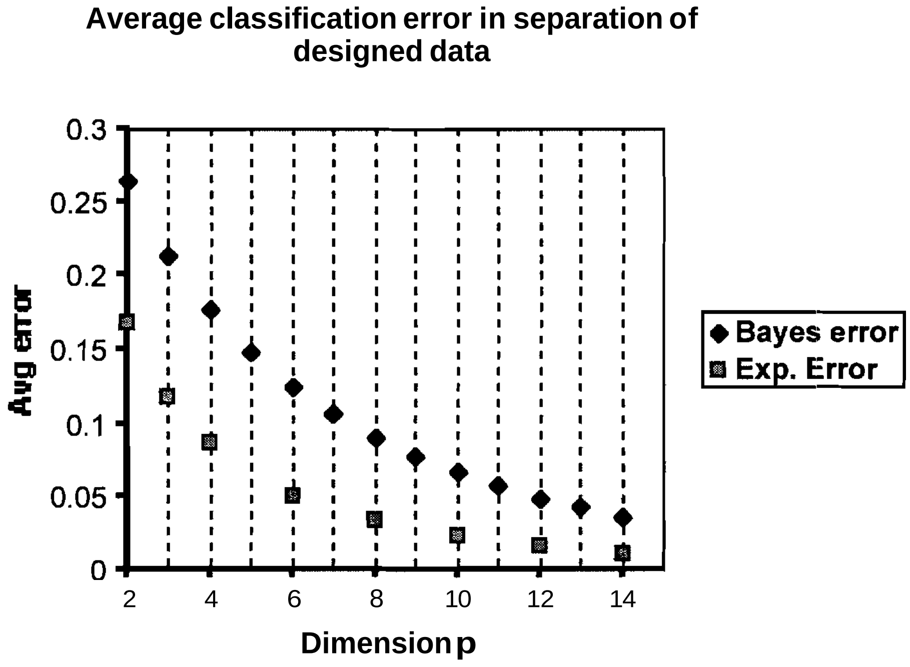


Fig. 4.13: Classification performance on experimental data. 'Bayes error' plots the theoretical best-case performance of analysis without use of spatial information. 'Experimental error' plots the results for the classification performance, averaged over 15 test runs for select values of p, for analysis using the 2×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=$1.5\chi_2^2$+ 2ln 2.
- f_r2 plots the density function for r_2=$0.375\chi_2^2$+ 2ln 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).

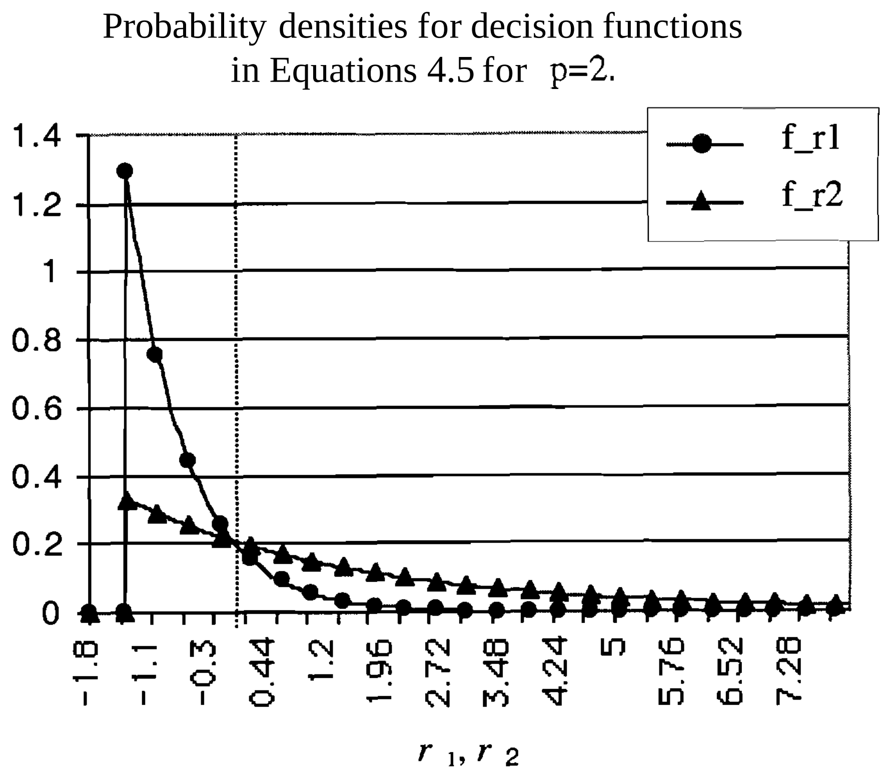


Fig. 4.14: Plots for the probability density functions for the operands of the decision rules in Equations 4.5, for p=2.

Recall the Bayes decision rule for separating the designed data.

$$r(y) = \frac{3}{800}(y-\eta)^t(y-\eta) + \frac{1}{2}\ln\frac{1}{16} \underset{I}{\overset{II}{\gtrless}} 0$$

The incorporation of the spatial model biases the decision rule[8] as below.

$$r(y) = \pm 2q + \frac{3}{800}(y-\eta)^t(y-\eta) + \frac{1}{2}\ln\frac{1}{16} \underset{I}{\overset{II}{\gtrless}} 0 \qquad (4.7)$$

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,), 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 AE 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 $1.5\chi_2^2$+ 2ln 2, $1.5\chi_2^2$+ 2ln 2+2q, and $1.5\chi_2^2$+ 2ln 2-29, according to the bias applied to the decision rule.
- The error in classification of D, 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 -29, 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 $1.5\chi_2^2$+2ln 2+2q to the right of the decision threshold.

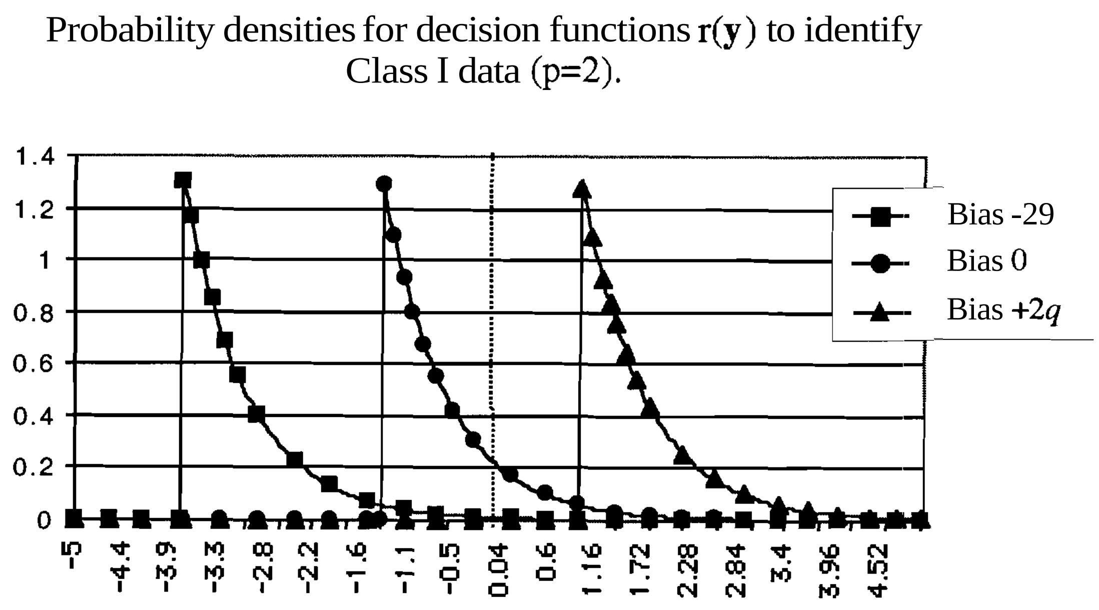


Fig. 4.15: Density function plots for $1.5\chi_2^2$+ 2ln 2 + {0, -2q, 2q}. Note that the area under the curves to the right of the decision threshold (shown as the vertical line) corresponds to the respective errors in classification of the data D,.

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.

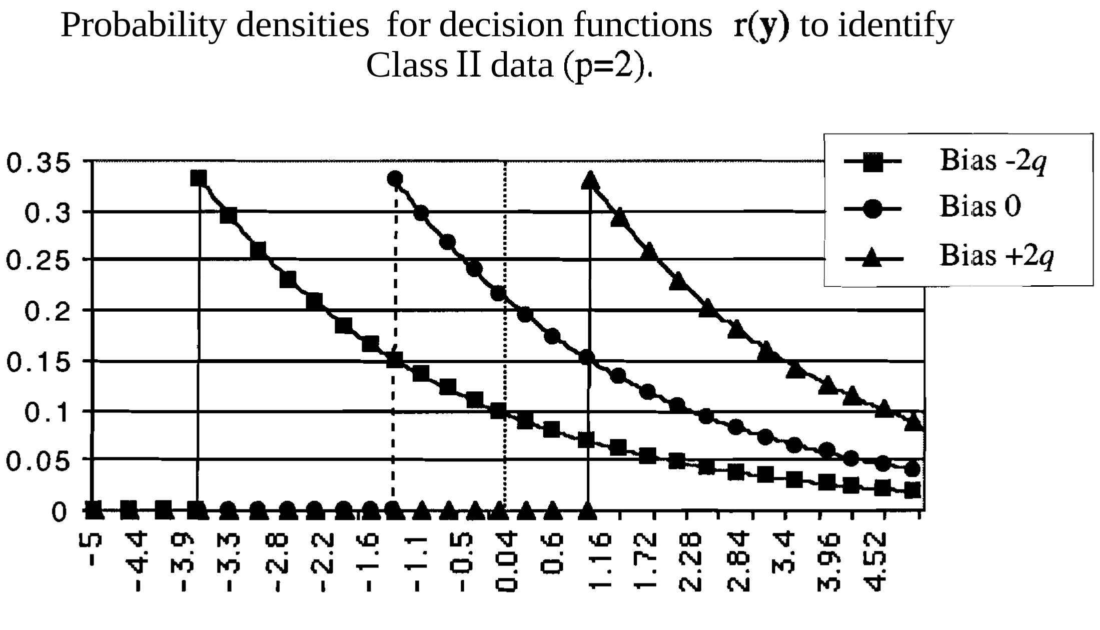


Fig. 4.16: Density function plots for $0.375\chi_2^2+ 2\ln 2 + \{0, -2q, 2q\}$. Note that the area under the curves to the left of the decision threshold (shown as the vertical line) corresponds to the respective error in classification of the data D,.

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.

■ Class I
Class II

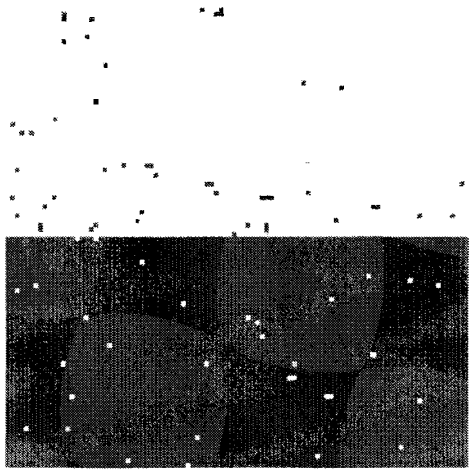


Fig. 4.17: A sample classification output of the proposed algorithm implemented on the designed data.

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 –

  $r_1=1.5\chi_p^2+p \ln 2$, and $r_2=0.375\chi_p^2+ \text{p} \ln 2$
- Errors in classifying D, 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 classification map, and a sample representation of the algorithm output (with misclassifications).

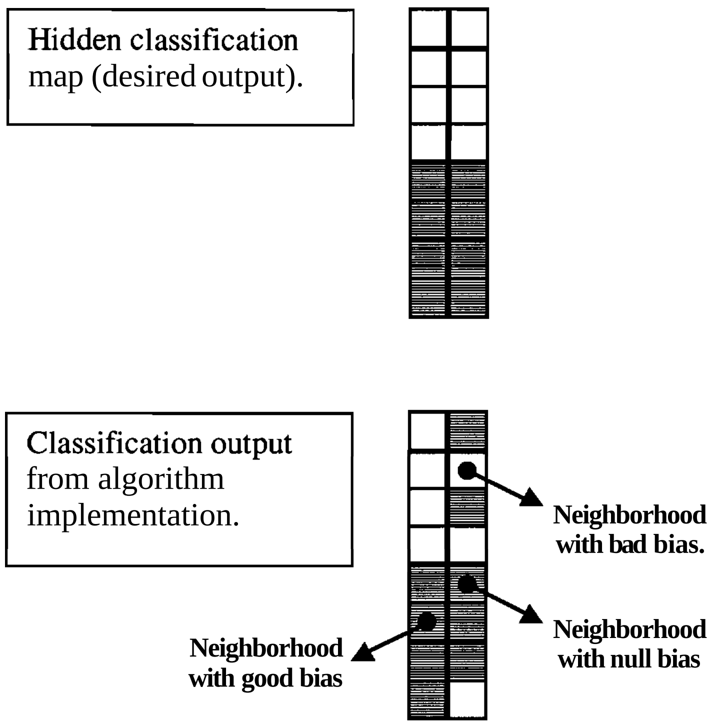

Hidden classification map (desired output).
Classification output from algorithm implementation.
Neighborhood with bad bias.
Neighborhood with null bias
Neighborhood with good bias

Fig. 4.18: Illustration of the influence of the neighborhood in biasing the decision rule.

Note that scene correlation is a function of transitions observed in a sample of the image. For the designed 100×100 image, the number of transitions or flip-count can be calculated from the observed scene correlation as (refer Appendix A.7 for details)

$$\text{flip-count} = \text{round}[2 \bullet 99 \bullet (1 - \text{correlation})].$$

Based on this estimate, the error can be apportioned to the good-bias, the no-bias, and the bad-bias decision functions. The rule designed for the distribution takes the following logic -

- For the ideal classification, the pixels running horizontally along the center of the image receive no bias. The proportion of the pixels in the image that receive bias is thus, 98%.
- There are 2•99 - 1 potential transitions in the vertical strip a hundred pixels long. Of these, the ideal classification map sees a single transition, for the designed experiment.

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

$$\text{bad\_ratio} = \frac{1}{2}\left(\frac{\text{flip\_count} - 1}{2 \bullet 99 - 1}\right).$$

- '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_{er},' is calculated as

  0.98•(good-ratio•P_{err}^{I}_with_good_bias+bad-ratio•P_{err}^{II}_with_bad_bias)

  + 0.02•P_{err}^{I}_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 $P_{„,}^{I}$ using the values from Table 4.3, can be re-calculated as

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

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

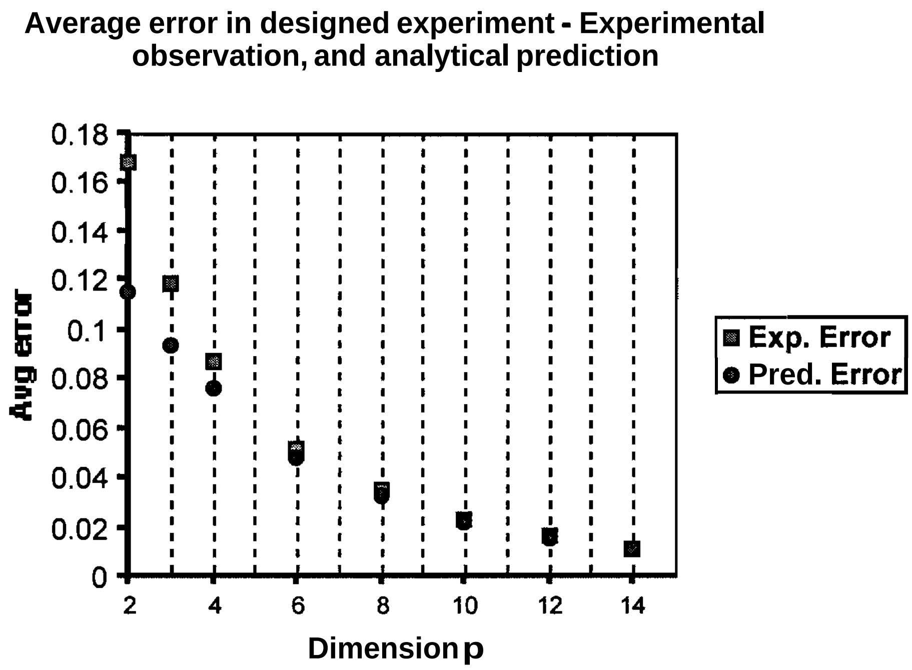


Fig. 4.19: Plot of classification errors in separation of designed data - experimental values and analytical predictions.

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 ym.). 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.

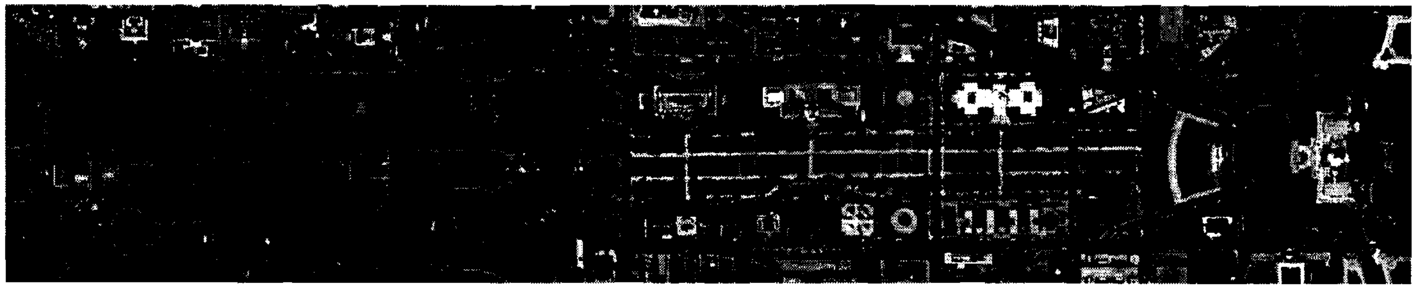


Fig. 4.20: Three color representation of DC flightline

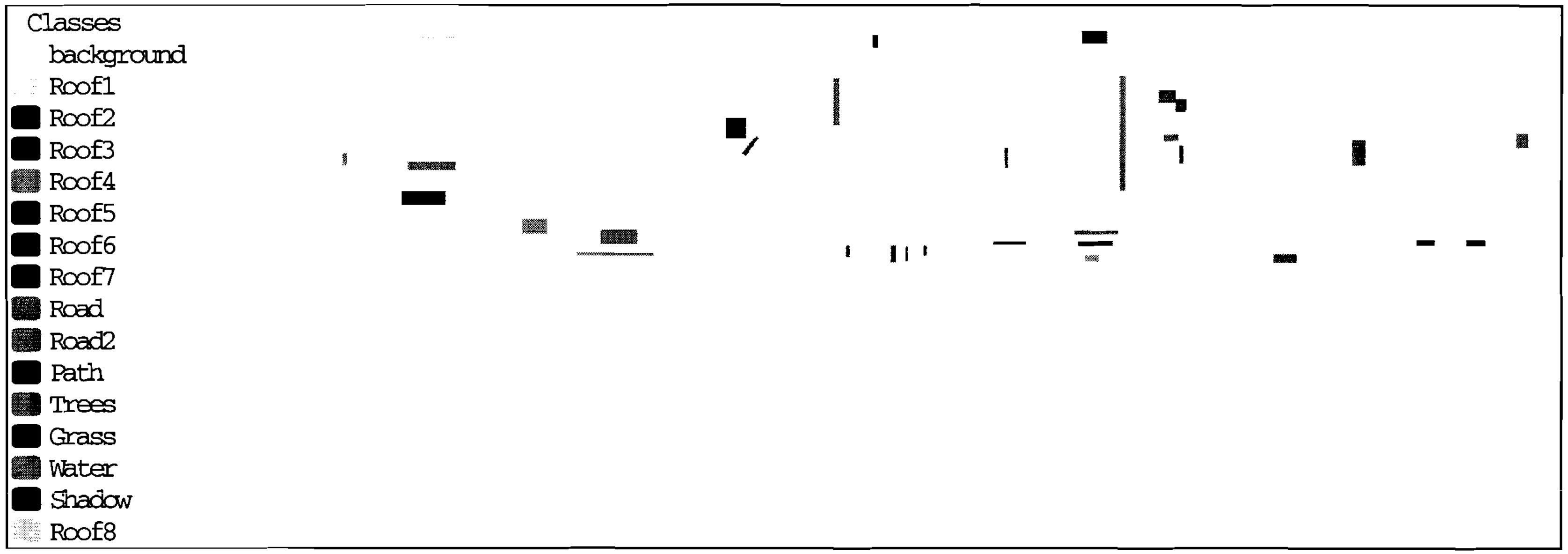


Fig. 4.21: Representation of training fields used in maximum likelihood classification of the DC flightline hyperspectral data [63].

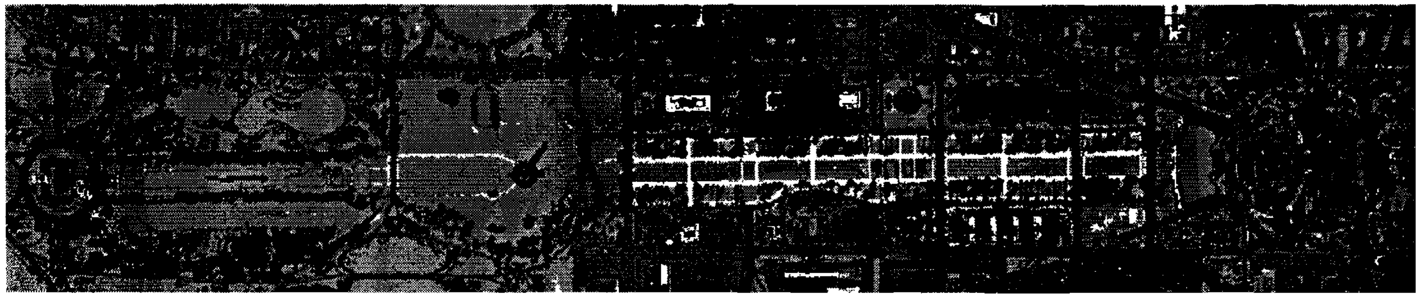


Fig. 4.22: Representation of classification output using training data shown in Figure 4.21. Note the confusion in distinguishing GRASS (in light green) from TREES (in dark green) in the section at the far right.

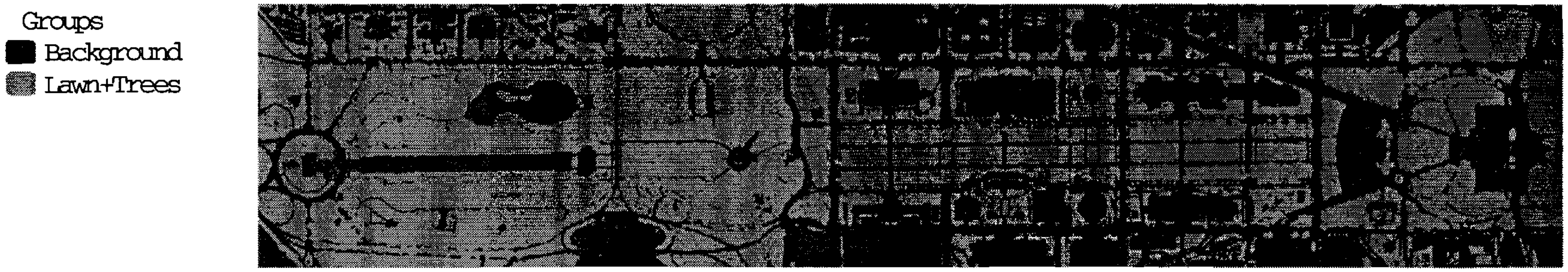


Fig. 4.23: Representation of data isolated as either class LAWN or TREES (in green) as produced in the hierarchical scheme of analysis.

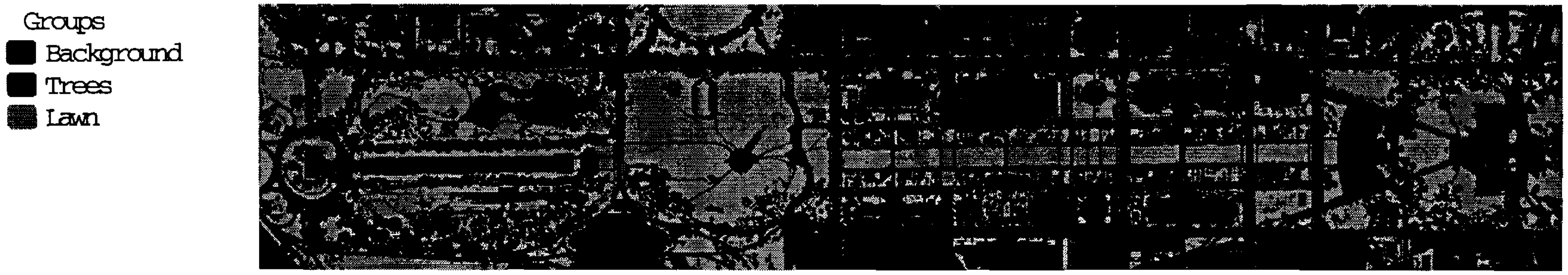
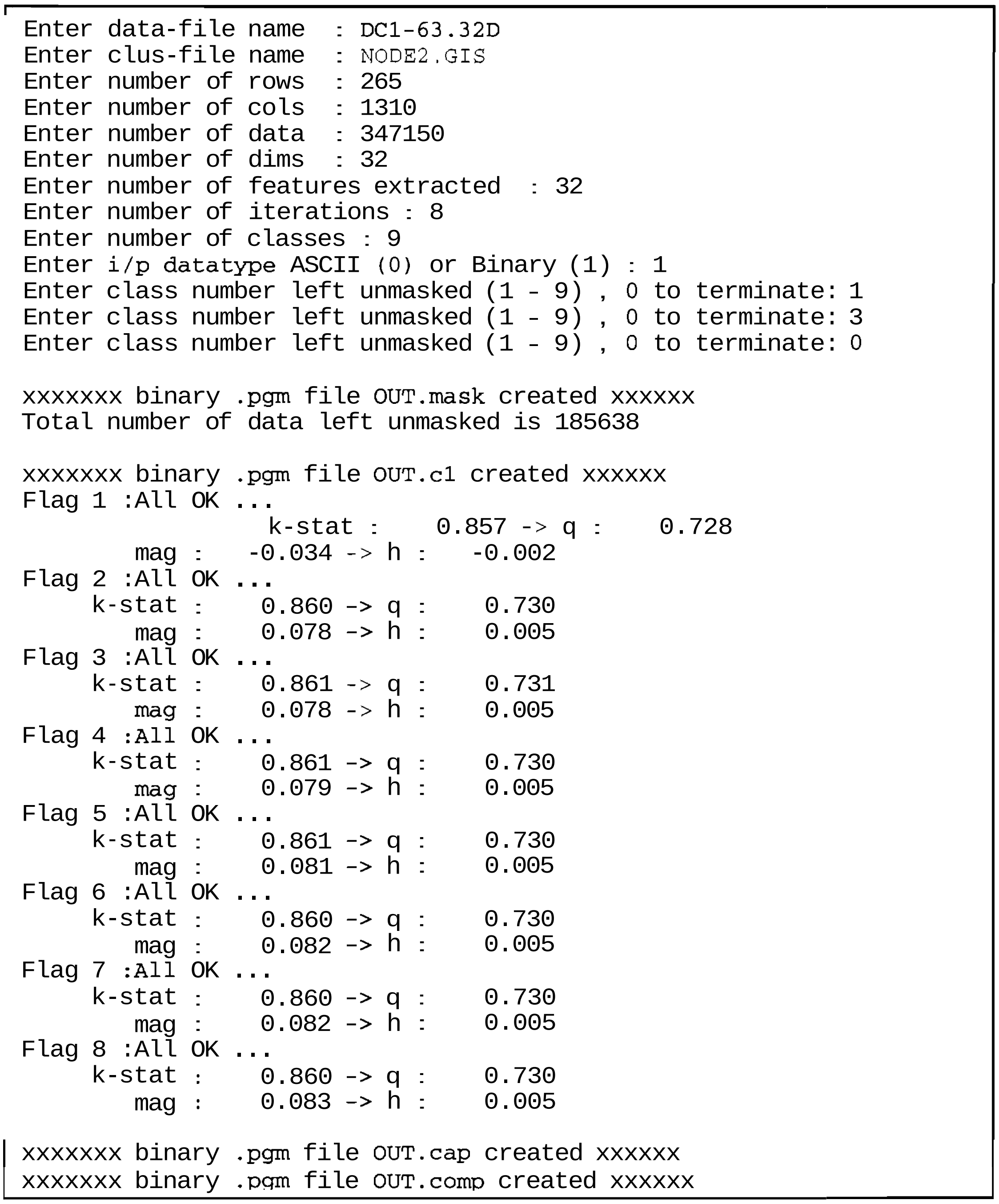


Fig. 4.24: Output of the spectral-spatial scheme with the $2\times N$ model used for the separation of classes LAWN and TREES.



```
Enter data-file name  : DC1-63.32D
Enter clus-file name  : NODE2.GIS
Enter number of rows  : 265
Enter number of cols  : 1310
Enter number of data  : 347150
Enter number of dims  : 32
Enter number of features extracted  : 32
Enter number of iterations : 8
Enter number of classes : 9
Enter i/p datatype ASCII (0) or Binary (1) : 1
Enter class number left unmasked (1 - 9) , 0 to terminate: 1
Enter class number left unmasked (1 - 9) , 0 to terminate: 3
Enter class number left unmasked (1 - 9) , 0 to terminate: 0

xxxxxxx binary .pgm file OUT.mask created xxxxxx
Total number of data left unmasked is 185638

xxxxxxx binary .pgm file OUT.c1 created xxxxxx
Flag 1 :All OK ...
                 k-stat :    0.857 -> q :    0.728
        mag :   -0.034 -> h :   -0.002
Flag 2 :All OK ...
     k-stat :    0.860 -> q :    0.730
        mag :    0.078 -> h :    0.005
Flag 3 :All OK ...
     k-stat :    0.861 -> q :    0.731
        mag :    0.078 -> h :    0.005
Flag 4 :All OK ...
     k-stat :    0.861 -> q :    0.730
        mag :    0.079 -> h :    0.005
Flag 5 :All OK ...
     k-stat :    0.861 -> q :    0.730
        mag :    0.081 -> h :    0.005
Flag 6 :All OK ...
     k-stat :    0.860 -> q :    0.730
        mag :    0.082 -> h :    0.005
Flag 7 :All OK ...
     k-stat :    0.860 -> q :    0.730
        mag :    0.082 -> h :    0.005
Flag 8 :All OK ...
     k-stat :    0.860 -> q :    0.730
        mag :    0.083 -> h :    0.005

xxxxxxx binary .pgm file OUT.cap created xxxxxx
xxxxxxx binary .pgm file OUT.comp created xxxxxx
```

Fig. 4.25: Text output of program implementing the proposed scheme for spectral-spatial analysis on the DC data. The algorithm is presumed to have converged when successive iterations return little or no change in estimated values of q and *h*.

4.5.2 Forest data

This data was also collected using a HYDICE scanner over 210 spectral channels. However, for the purpose of this analysis, only six channels were chosen for post-processing of the initial classification obtained by [64]. A 200x200 section of the data was selected, and the class SHADOW as identified through maximum likelihood classification (cf. Figures 4.26-27) was segmented into two distinct classes. The results of the experiment are shown in Figure 4.28. Of note is the discovery that the class SHADOW in the original segmentation actually comprises two spectrally distinct sub-classes - corresponding to shadows cast by trees on grass, and shadows cast by trees on other trees.

The text output of the program is included in Figure 4.29.

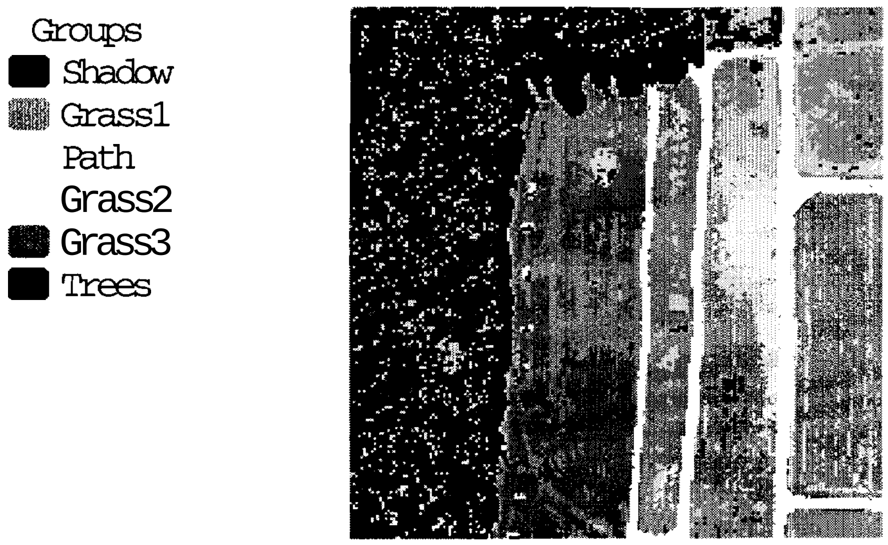


Fig. 4.26: Representation of maximum likelihood classification output of analysis on Forest data [64].

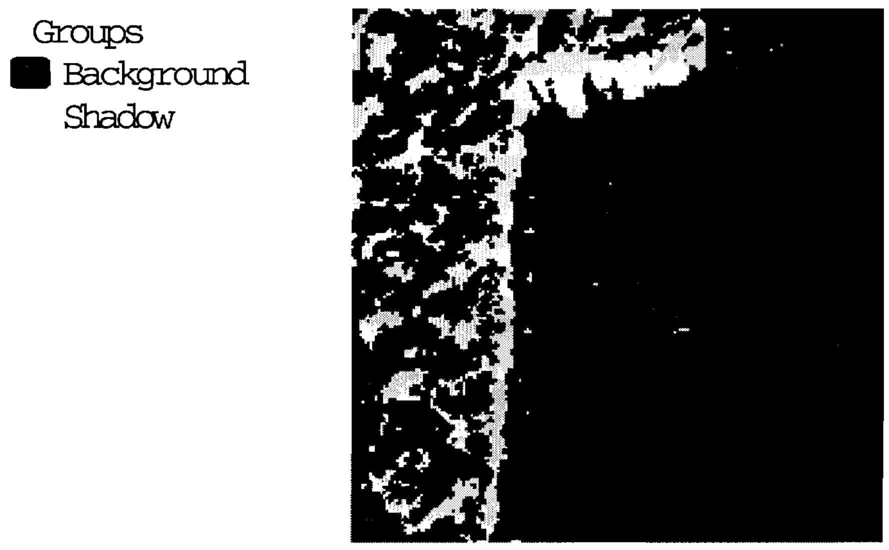


Fig. 4.27: Class SHADOW isolated from the output in Figure 4.26

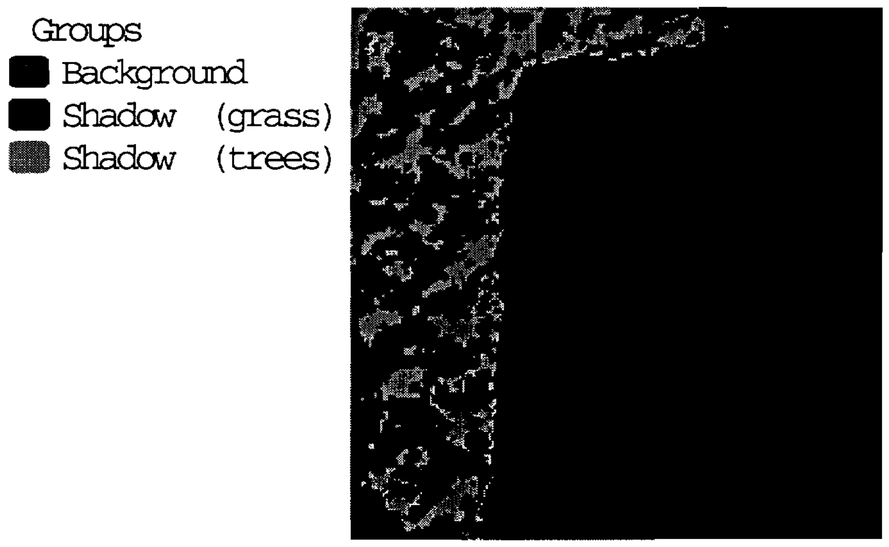


Fig. 4.28: Segmentation of the isolated class SHADOW from Figure 4.27 into sub-classes using the spectral-spatial analysis. The separation into distinct sub-classes highlights the distinction between shadow-on-grass and shadow-on-trees.

```
Enter data-file name  : HDC_data
Enter clus-file name  : HDC_clus
Enter number of rows  : 200
Enter number of cols  : 200
Enter number of data  : 40000
Enter number of dims  : 6
Enter number of classes : 9
Enter number of iters   : 10
Enter choice - Regular (0) or Advanced (1) : 1
Enter i/p datatype ASCII (0) or Binary (1) : 0
Enter class selection (1 - 9) : 2
Select another class, or 0 to ignore : 0

xxxxxxx binary .pgm file OUT.mask created xxxxxx
Total number of data left unmasked is 4855

xxxxxxx binary .pgm file OUT.c1 created xxxxxx
Flag 1 : ALL OK ...
     k-stat :    0.754 -> q :    0.710
        mag :    0.000 -> h :    0.000
Flag 2 : ALL OK ...
     k-stat :    0.895 -> q :    0.812
        mag :    0.172 -> h :    0.009
Flag 3 : ALL OK ...
     k-stat :    0.930 -> q :    0.905
        mag :    0.212 -> h :    0.011
Flag 4 : ALL OK ...
     k-stat :    0.945 -> q :    0.955
        mag :    0.227 -> h :    0.011
Flag 5 : ALL OK ...
     k-stat :    0.953 -> q :    0.964
        mag :    0.239 -> h :    0.012
Flag 6 : ALL OK ...
     k-stat :    0.956 -> q :    1.003
        mag :    0.249 -> h :    0.012
Flag 7 : ALL OK ...
     k-stat :    0.959 -> q :    1.008
        mag :    0.262 -> h :    0.013
Flag 8 : ALL OK ...
     k-stat :    0.961 -> q :    1.010
        mag :    0.264 -> h :    0.013
Flag 9 : ALL OK ...
     k-stat :    0.961 -> q :    1.011
        mag :    0.268 -> h :    0.013

xxxxxxx binary .pgm file OUT.cap created xxxxxx
xxxxxxx binary .pgm file OUT.comp created xxxxxx
```

Fig. 4.29: Text output of program implementing the proposed scheme for spectral-spatial analysis on the Forest data. The algorithm is presumed to have converged when successive iterations return little or no change in estimated values of q and ***h.***

PART TWO

5. INTERPRETING REMOTE SENSING DATA

Part Two of this dissertation proposes a process model for the analysis of remote sensing data. If the input to the analysis is scanner data, and the output is knowledge - the key to a successful extraction of knowledge from the data is information provided by the analyst. As an example, the visual representation of satellite data provides insights into the data that are invaluable to hyperspectral analysis [2]. While the design of such analyses may appear task specific, it is believed that the analytical process possesses an unchanging modular structure that can be framed as a model. The elaboration of this model is the goal of this work.

In the above context, the analyst is considered a source of data, and the process of analysis is a data fusion. The phrase 'data fusion' has been interpreted differently in different research. For the present discussion, the definition by Wald [65] is highlighted below as being of greatest relevance -

"... data fusion is a formal framework in which are expressed means and tools for the alliance of data originating from different sources".

Finally, it should be noted that the ensuing discussion does not attempt a design of learning/intelligent algorithms. The emphasis lies on the development of a synergy between the human and the computer, with a judicious utilization of the strengths of the respective sources in the analysis.

Sections 5.2-3 discuss a process model for the analysis of remote sensing data, and Chapter 6 presents a case study incorporating the model. The data used in the analysis is the D.C. data-set of Section 4.5.1, and the task is that of the classification of the data into the relevant scene-classes.

5.1 Previous Work

In the context of remote sensing analysis, data fusion is probably most commonly encountered as sensor fusion - the combination of data from various scanner-sources over a given scene. This may be attributed to the tendencies of various remote sensing consortia around the world to emphasize scanner technologies of relevance to their respective needs and geographical locations. For example, the French Satellite Pour 1' Observation de la Terre (SPOT) has high-resolution panchromatic scanners, the Canadian RadarSat gathers data in the microwave region, and the U.S. Landsat carries the Multispectral Scanner (MSS) that emphasizes the visible and near infra-red regions of the spectrum [1]. While the respective scanners have attributes peculiar to their usage, there has been considerable interest in the fusion of their outputs. F'or instance, the panchromatic data is usually gathered at a high resolution, and it is possible to carry out a filtering operation to obtain the high spatial frequency characteristics in the image. If this information is coupled with that obtained from multispectral data, the analyst can obtain a clear demarcation of objects in the scene that comprise spectrally similar pixels. Such an approach yields results that have ready visual interpretation, cf. [66] [67]. Further enhancements to this approach to sensor fusion have been developed using the Wavelet transform [68] [69].

Another route to data fusion is that of 'combination of perspectives'. In other words, the interpretations of the scene produced by different models are merged to yield a composite that is presumed to be more robust against sensor noise or algorithmic deficiencies. Representative work in this direction is that of [70] via decision trees, and that of [71] via genetic algorithms. [72] regard such a design as a fusion of concepts, and present a theory for the combination of N-learners fusion.

Data fusion has also been discussed at various abstract levels. Consider the distinction between computer-based analysis and human thinking at the logical level. While computer programs process information with binary (Yes or No) logic, human decision-making employs various degrees of belief. Mathematical research in modeling the latter has led to various theories on measures of belief. Some examples are the Dempster-Shafer evidence theory, and fuzzy logic theory. Bloch [73] has conducted a review of these in the context of data fusion.

Perhaps the best example of data fusion in everyday life is human reasoning. The ability of the human to process diverse data into coherent decision-making is remarkable. Consequently several researchers have devoted energies to the modeling of systems that copy human thinking for 'intelligent' decision making. However, even before an intelligent solution can be designed, the problem and the data have to be represented in forms suitable for analysis. This is not a trivial task. Some research of relevance are [74] [75] [76]. [77] discuss fusion at the query level, and use logical operators for the fusion. Although the experimentation does not appear directly relevant to the analysis of remotely sensed data, such research is important in that it reveals the thinking of the human in the dissection of a problem and the design of its solution.

As opposed to the design of intelligent algorithms in data fusion, some recent work has focused on developing process models that can guide analysis. The philosophy of this approach is well described in a work by [78] in the context of military strategizing. The paper highlights the division of responsibilities between person and machine. The authors state that the task of making tactical decisions in naval operations is too complex to be accomplished by humans alone or by computers alone, ancl present several examples in support of the statement. They claim that the human uses judgment and native intuition to make decisions, whereas the assessment of the physics of the situation is a highly mathematical endeavor best left to the computer.

A fairly comprehensive discussion of the process model for data fusion is contained in various articles in the January **1997** issue of the Proceedings of the IEEE. Some notable research is described here. [79] present an overview of the subject, discuss potential applications and present a taxonomy; [80] presents various fusion architectures classified on the basis of I/O and variants thereof; [81] takes the perspective of a database specialist, and discusses the role of database management in data fusion. [82] summarizes these works and also presents a probability scheme for weighting the distributed outputs from various decisions for combination into a single composite decision. While such discussions are valuable, they are incomplete for want of definitive examples demonstrating the theory. The omissions are probably not accidental on account that most of the represented research is supported by United States defence organizations. In this regard, the research presented in this dissertation is an important contribution in that the process model is justified by its successful application to a practical problem in remote sensing data analysis.

The May 1999 issue of the IEEE Transactions on Geoscience and Remote Sensing is devoted exclusively to research on practical implementations of data fusion. Of special note is the work of [83] for cartographic feature extraction. 'The methodology and the principles guiding the analysis are similar to those for the case study elaborated in Chapter *6*.

5.2 Principles of Data Fusion

The motivation for instituting a set of rules for remote sensing data analysis stems from the highly complex and dynamic nature of the scene whose understanding is desired - the Earth's surface. Laboratory models for terrestrial phenomena are usually successful only to the extent of the span of a very limited set of observations. The ability of the human to learn and adapt analysis to the peculiarities of the problem, is thus invaluable. The elaboration of a few, basic principles of data fusion is intended to guide analysis, and maximize the benefit of human-machine interaction.

On the other hand, mathematical modeling on the computer serves a useful purpose, in that the system dynamics can be reduced to the manipulation of a few parameters. If applicable, the complexity of the ensuing analysis can be significantly reduced, and thus be appropriated by the user into a suite of analysis-routines. The latter point is emphasized since the complexity of the processing algorithm and the associated performance has to be balanced by the algorithm's understanding and acceptance by the user. The conclusion being that, when optimizing human-machine interaction, ergonomics should be a design consideration.

<u>Axiom 5.1</u>: Human abilities are different from those of the computer.

Consider Table 5.1 overleaf, adapted from Schniederman [84].

Table 5.1
The human versus the computer in data analysis.

Human	Computer
Can draw upon experience and adapt decisions to unusual phenomena.	Can perform repetitive pre-programmed actions.
Can reason inductively, and process hierarchically.	Can process several items simultaneously.
Can generalize from observations	Can implement the generalizations.
Output depends on goal interpretation.	Output conforms to doctrines and performance indices - as determined by goal interpretation.
A source of data/information.	Has short response time, high speed of computation, and cheap data storage.

The conclusion drawn from Table 5.1 is that the inferential aspects of the analysis are best relegated to the human. The computer's abilities lie in the implementation of the schemes (of analyst design). The goal of this work is not to develop schemes that duplicate human behavior, but to identify rules the use of which optimizes machine-human interaction for superior performance in the task at hand.

<u>Axiom 5.2</u>: The machine validates what the user suggests.

In various applications, the output of the algorithm is a measure of belief in the hypothesis posed by the analyst. However, a poor output does not necessarily imply algorithmic deficiencies. Failure can be a result of the performance index being inadequate to the target task. An analysis is usually directed by the optimization of a user-defined performance measure. Incompatibility between this measure and the objective is unlikely to produce the desired results. In regard to analyses that seek a visual interpretation of the data, this is an especially important (and often overlooked) issue. Algorithms that process data through the optimization of mathematical criteria are often sub-optimal in the sense that the output image is cluttered (or fuzzy or noisy) and is visually unpleasing.

Axiom 5.3: Every analysis requires at least one revision.

The current level of technology precludes the possibility of an iintelligent system that operates independent of human input. Usually, analysis comprises various algorithmic 'objects', selected from a suite of procedures, linked in the appropriate sequence by the analyst. The optimal selection and ordering of these objects is often not known. Occasionally, algorithm parameterization is also dependent on human input. It may thus be concluded that most any test run of the process is likely to produce results that can be improved upon through experimentation.

It may also be inferred from the above axiom that a readily interpretable form for the application output is highly desirable.

Definition 5.4: Data fusion is an interface that allows collaboration among data sources to execute an analysis, enabling an assessment that is superior to one in which the sources are incorporated singly.

The above definition is an enhancement to the proposition of Wald [65] under the belief that definitions of the task and of the associated performance criterion are critical to the process of data fusion.

At this point some examples of data fusion are considered.

- The Gaussian model: Multispectral data can be looked upon as measurements on different sensors mapped to disjoint frequency bands. An efficient procedure for analyzing such data is to apply the Gaussian assumption, and assume that each vector is an observation on a multivariate Gaussian process. The vector elements can then be analyzed concurrently, and fusion is implicit. This model has been used successfully in various remote sensing data applications [35] [1].
- Bootstrapped averaging (bagging): This technique has been popularized by Breiman [85]. Multiple bootstrap samples are drawn from the data, and a different model is constructed for each. The results are then combined by simple averaging.

- Decision tree classification: This is an intuitive, and highly popular, scheme for the classification of data [86] [42] in a hierarchical fashion. Every node of the tree is an analytical model that separates the input data into distinct clusters each of which form input data to nodes at the next level. The process is continued to the extent of separation desired.

The following proposition is now stated without proof.

Proposition 5.5: Data fusion can be performed pre- or post- analysis. The performance of the former scheme, if implementable, exceeds that of the latter.

Proposition 5.5 is an attempt at a taxonomy for data fusion algorithms. In the example of the Gaussian model, it is evident that fusion is taking place prior to analysis. Statistical analysis of multispectral data requires the computation of the mean and the covariance per scene-class [2]. The latter set of statistics is in fact, a quantitative representation of the interaction among data collected at different wavelengths. The model is thus an example of data fusion, as achieved through the second order statistics. Of note is that the fusion is implicit in the input to the analysis. The class of schemes that achieve pre-analysis fusion do so through the assumption of a model that links the sensor data by an analytic process. Such schemes shall subsequently be referred to as CLASS I - see Figure 5.1.

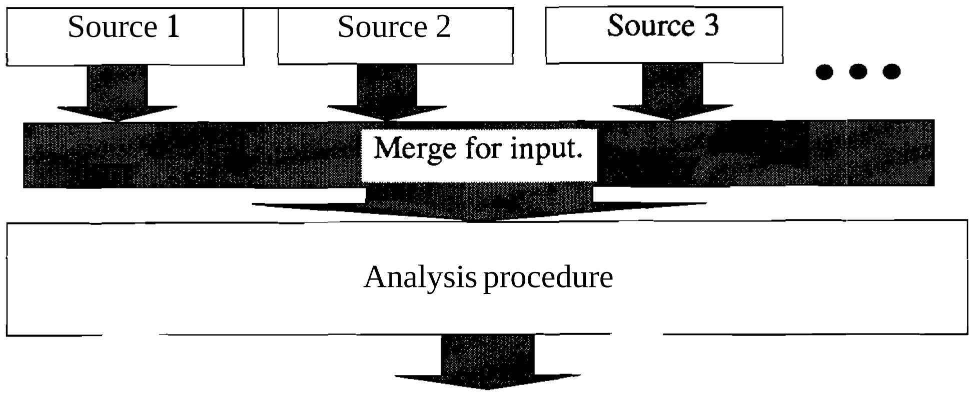


Fig. 5.1: CLASS I fusion - Data fusion applied at the input level prior to analysis (e.g. Gaussian model, lattice model).

The example of the 'bagging' [85] scheme is that of a post-analysis data fusion (henceforth referred to as a CLASS II type scheme) - see Figure 5.2. The data from various sensors are input separately to different analyses. The respective outputs are then merged separately as the next stage, to yield an output that is believed to be robust against data corruption and systemic errors. Herein lies another distinction from Class I type schemes in that the former class of analyses is geared towards improving classification accuracy as opposed to robust decision-making. Other examples of CLASS II data fusion are voting schemes and genetic algorithms.

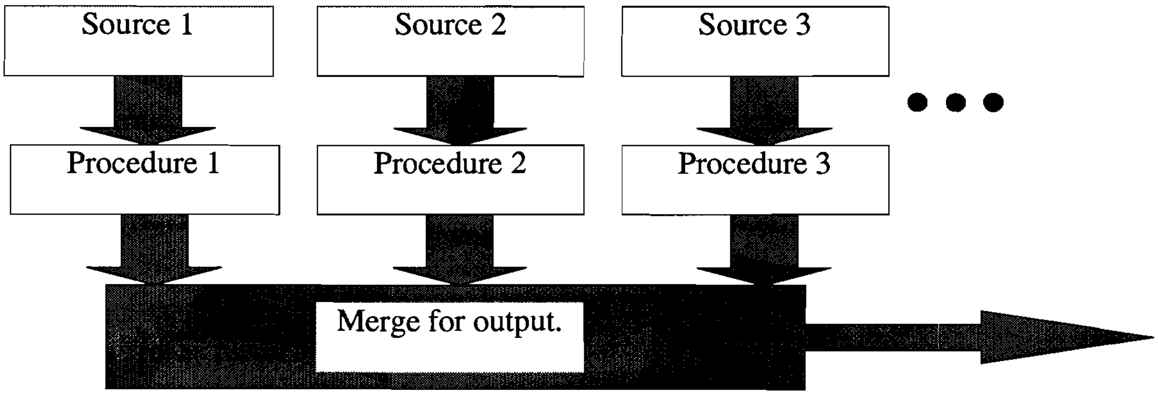


Fig. 5.2: CLASS II fusion - Data fusion applied post analysis at the output level (e.g. bootstrapped averaging [85]). Note that every 'procedure' block above, however primitive, is an example of a CLASS I fusion.

Often, analysis is a combination of modules of the two classes of data fusion schemes. The complete analysis is thus a hybrid - see Figure 5.3 overleaf. A good example of such a hybrid scheme is the decision tree [41][42].

The claim that CLASS I fusion is superior to CLASS II fusion is a conjecture based on experimental evidence. For instance, the statistical analysis of multispectral data can proceed without incorporation of covariance information, in which case, fusion occurs post-analysis. It is the use of second order statistics that elevates the analysis to a CLASS I type fusion. Experiments by [87] have demonstrated that the usage of class-covariance information can significantly improve hyperspectral analysis provided the statistics can be accurately estimated.

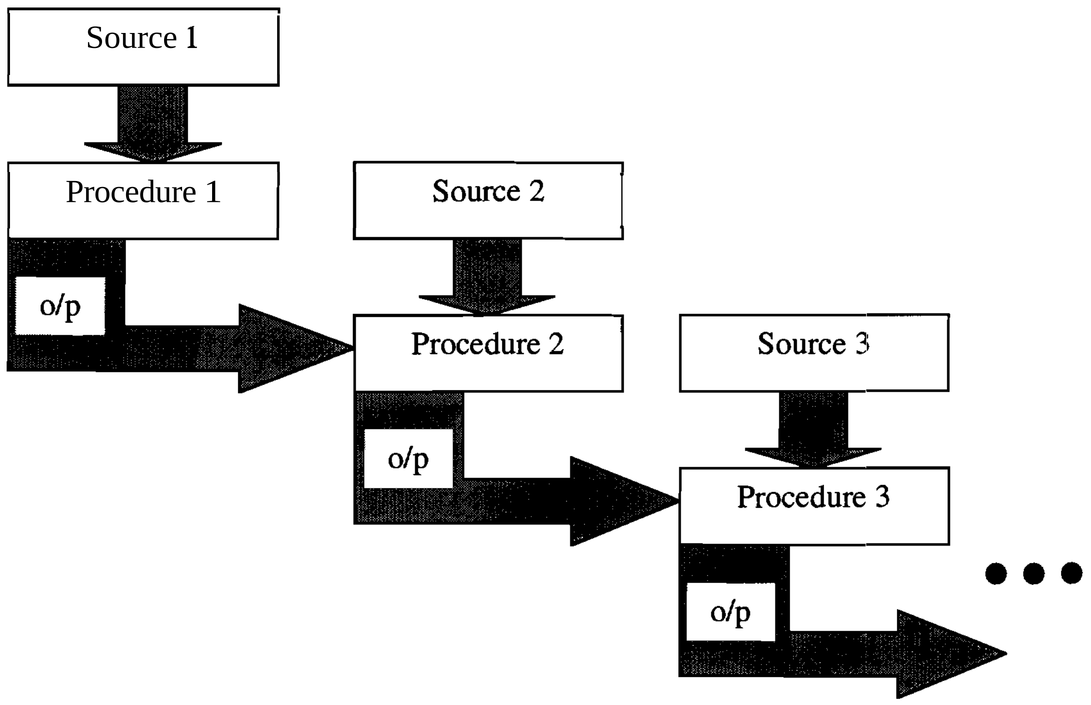


Fig. 5.3: Hybrid scheme of data fusion with output of one level input to another (e.g. decision tree methodology).

5.3 A Process Model

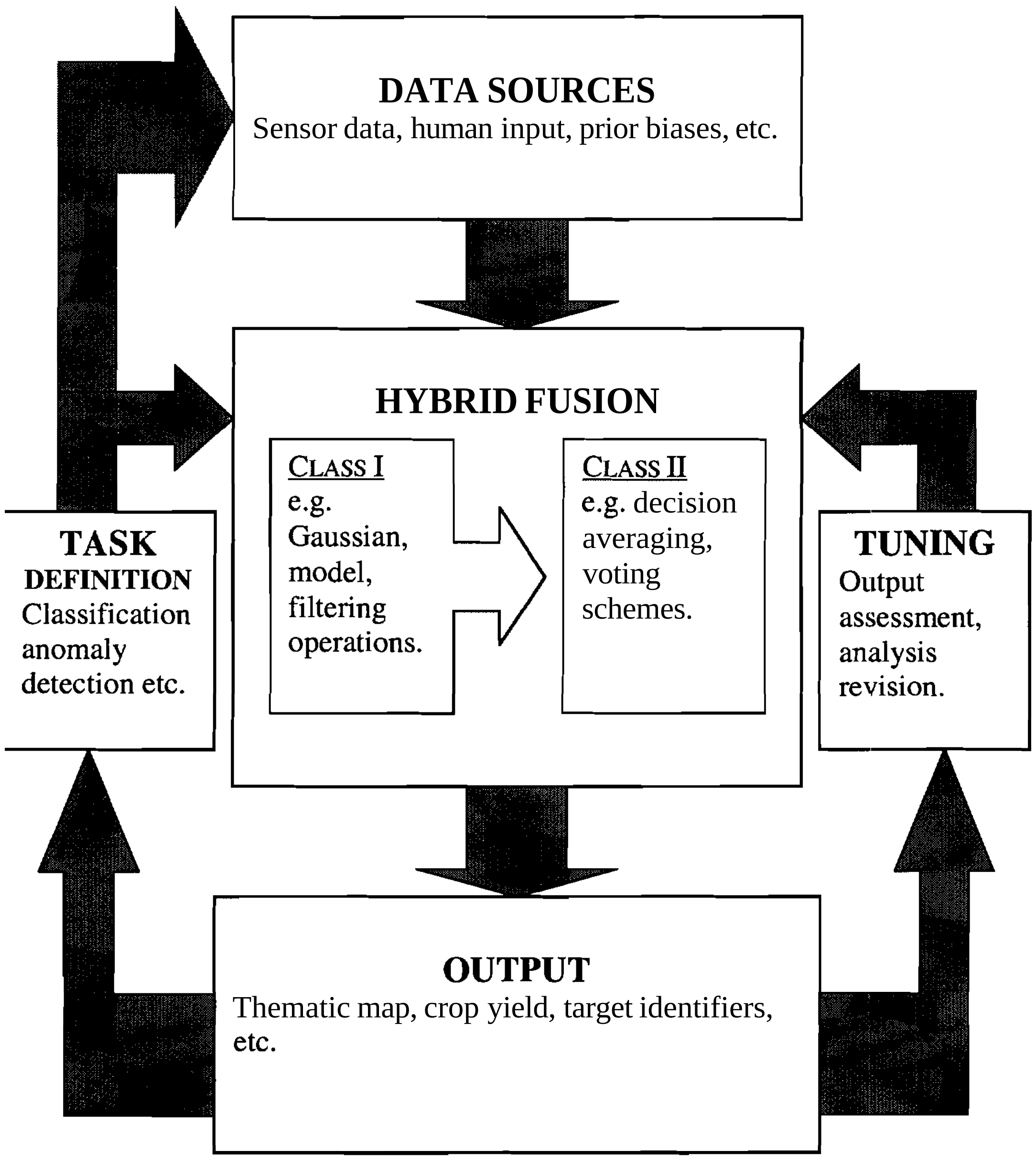

DATA SOURCES
Sensor data, human input, prior biases, etc.
HYBRID FUSION
CLASS I
e.g. Gaussian, model, filtering operations.
CLASS II
e.g. decision averaging, voting schemes.
TASK DEFINITION
Classification anomaly detection etc.
TUNING
Output assessment, analysis revision.
OUTPUT
Thematic map, crop yield, target identifiers, etc.

Fig. 5.4: Process model for remote sensing data analysis.

The schema of Figure 5.4 is an intuitively designed procedural guide for data fusion. The modules of the schema are explained below.

- TASK DEFINITION - Every engineering problem can be designed a solution provided the task is well defined. A rigorous task definition enables a judicious selection of the data sources, and a design of the analysis procedure.
- DATA SOURCES - The data sources, though restricted by availability, need to be assessed for their utility towards the task at hand, and for the presence of noise. The 'scrubbing' stage, as discussed in data-mining [88] literature, is incorporated here.

DATA ANALYSIS- The proposed definition of fusion encompasses all techniques of data assessment. Given the task, a CLASS I fusion scheme is used to process the input data, and condense information that can be utilized in a CLASS II type mechanism. The complete structure is thus a hybrid, as per the discussion of the previous section. Design considerations include procedural robustness and a well-designed performance criterion (Axiom 5.2).

OUTPUT - The output of the analysis needs to be readily interpretable by the analyst (Axiom 5.1). A visually accessible output is especially desirable. The case study presented in the next section will illustrate the value of image representations in remote sensing data analysis.

- TUNING - Finally, once the output has been assessed, the analyst proposes procedural modifications (Axiom 5.3) and redoes the process.

6. A CASE STUDY - ANALYZING THE D.C. FLIGHTLINE

The HYDICE scanner gathers data over 210 channels (samplings of the energy spectrum) for a region 1310 elements x 265 elements in size. It is believed that the scene is constituted of specific scene-classes (such as water, grass, trees, road etc.). The identification of a set of scene-classes that comprehensively describes the scene is central to this dicussion, and is one of the objectives of this analysis. The spectral data measures the energy reflected off the Earth's surface. Thus, it is believed that the spectral measurements on a portion of the Earth contain the information from which the corresponding terrain-type or land-usage can be identified. The analysis processes the available data and assigns a label, from the set of scene-classes, to each of the data. A color representation of the output is also known as a thematic map or a classification map, the colors used in the representation being a one-to-one map from the set of scene-classes. The task may also be viewed as data compression or as knowledge extraction from the spectral data.

At this point, the process model of Section 6.2 is invoked to sketch the procedural design.

- DATA SOURCES - The primary source of data, other than the analyst, is the HYDICE scanner. An additional source of information is the Digital Elevation Map (DEM) of the scene. The DEM will be discussed later in Section 6.2.6 at the point of its usage.
- TASK DEFINITION - Before initiating the analysis, it is important to assess the task. The analysis requires the classification of the data into a set of scene-classes that, to the extent of the analyst's belief, spans the semantic content of the data. The given scene is of an urban nature, and one of the goals is the identification and demarcation of buildings in the scene. Given the aerial point of view of the remote sensing scanner, this goal may be re-stated as the demarcation of building roofs in the scene. A point of note is that data elements are better identified as

rooftops through their functional usage in the scene rather than through the material used in their construction (cf. Section 6.2.6 on rooftop extraction).

- DATA ANALYSIS - The various modules available for this analysis are discussed individually in Section 6.1 - The Fusion Suite.
- OUTPUT - As pointed out as an adjunct to Axiom 5.3, it is important to have an output that enables a quick and accurate assessment of the algorithm used in processing the data. (Past experience has shown Multispec [89] to be a software of immense utility in remote sensing data analysis, especially for data visualization.)

  TUNING - The tuning process is dependent on the stage at which the output is assessed. If the output is deemed to be deficient in a certain aspect, the analysis is modified appropriately and re-done. Alternately, a sub-routine may be custom designed to counter the problem. In general, it is believed that any engineering analysis requiring subjective evaluation can be solved to the extent desired, provided the terminal assessments can isolate the problem-spots for subsequent tuning.

A sequential scheme of modular design will be used in the analysis. Each stage in the analysis will be referred to as a node in the process-sequence. Human-computer interaction will be emphasized through the analysis, and the analysis at each node will depend on the subjective assessment of the output at the previous node. A quantitative assessment of the analysis will be presented in the synopsis of this chapter, Section 6.2.12.

6.1 The Fusion Suite

The various modules/methodologies used in this analysis are described below.

6.1.1 Maximum likelihood classification

The technique comprises the identification of training data -- data representative of each of the classes in the set of scene-classes – followed by the construction of decision rules for the classification of the data. The analyst input to the algorithm is the identification of a comprehensive set of scene-classes, and the selection of training data. The algorithm assumes the spectral data are observations on Gaussian processes whose parameters may be estimated using the training data. For a given element, the desired output is the identification of the generating process, for each of the data. Mathematical details on the technique can be found in [56] [2] [1].

6.1.2 Unsupervised segmentation using the lattice model

The technique was first presented in Chapter 4. For usage in this analysis, it is modified as per the discussion of Chapter 5.

An implication of Proposition 5.5 is that a supervised scheme of classification is potentially superior to an unsupervised one. The presentation of Chapter 4 was of a latter kind, the only input to the algorithm being the identification of the class(es) to be separated. It is surmised that superior performance may be obtained by directing the segmentation - the statistics for the classes into which separation is desired can be estimated beforehand and input to the algorithm. This is done by manually marking out training data in the scene, representative of each of the separated classes. The updating of the statistics through the iterations of the algorithm proceeds via the EM algorithm, cf. [90].

6.1.3 Fusion - HYDICE data + Digital Elevation Map @EM)

At a later stage, the HYDICE spectral data will be fused with the elevation map of the scene for the purpose of rooftop identification. While a Class I type fusion would be preferred, an appropriate design is not known. Consequently, a sequential Class II type (post-analysis) fusion is used. The method is presented in detail in Section 6.2.6.

6.1.4 Decision tree (or graph) structure

As per Axiom 5.3, it is believed that an acceptable solution does not take the form of a single algorithm. The optimal scheme for engineering a solution to the problem comprises an assessment-analysis cycle repeated as many times as desired. The analysis algorithm in each cycle can be modified as per the assessment of the previous stage. If the stages making up the final solution are laid out in the order of their occurrence, the proposed solution takes the form of a directed graph. A representation of the solution is shown at the end of the chapter in Figure 6.28.

6.1.5 Masking

As per the decision tree structure, at the end of a stage in the process sequence, it may be realized that portions of the data are accurately identified, while others are in error. The masking module isolates the data in error for subsequent processing, and excludes the remainder of the scene from analysis at the next stage. The selected data are said to be 'masked out' of the scene. Examples of the usage are in Section 6.2.3- Node 1 and Section 6.2.4 - Node 2.

6.1.6 Negative training

If the analyst has sufficient knowledge of the scene, he/she can conclude that a given scene-class is localized to a specific area. In such a case, all data outside the area that are labeled that scene-class, can be claimed rnisclassified. These data are then labeled a class other than the one under analysis. The ensuing analysis uses a simple re-labeling scheme - the scene-class of greatest frequency in the eight-point neighborhood of the rnisclassified element is chosen as the new class, provided that the new labeling itself is not in error. This process is termed "Negative training", and has been used in Section 6.2.7 - Node 5 and Section 6.2.8 - Nodes 6,7.

6.2 D.C. Flightline Analysis

6.2.1 Scrubbing - Removing bad data

The 210 spectral channels[9] of data need to be examined for corruption. Experience has shown that data can be corrupted for various reasons. Judgment on the: data per spectral channel can be made by visually examining the output for each [89]. Some causes for data corruption are listed below, with examples presented on the next page as Figures 6.1-2.

- Water absorption bands - an insufficient response over a given bandwidth. It can lead to an exaggerated sensitivity to sensor noise.

[9]The channel number, as in this usage, signifies a specific wavelength at which the spectrum of energy reflected off the Earth has been sampled. Correspondingly, the 210 channels for the HYDICE scanner are representative of samples at 210 distinct wavelengths.

- Physical defects in scanner, such as scratches.

Judgment on the data may be made by visually examining the output for each of the channels. Some examples of corrupted channels that were excluded from the data are shown in Figures 6.1-2. After excluding all such 'bad' data, 104 channels remained of the original 210 - Channels 55-100, 108, 113-132, 135, 152-164, 173, 180-201.

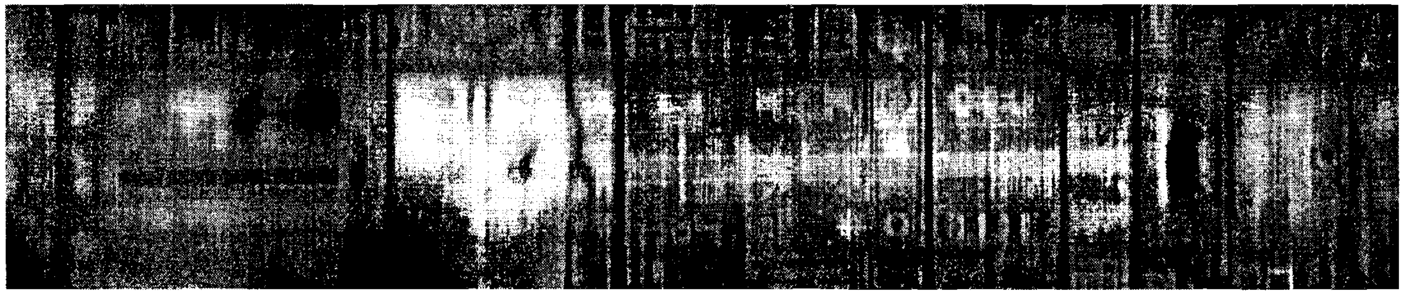


Fig. 6.1: Gray-level representation of data sampled at wavelength 1.37μm (Channel 105).

Fig. 6.2: Two-color representation of data sampled at wavelengths 2.43μm (Channel 202) and 2.48μm (Channel 207). Note the low energy of the response and the scratches running through the length of the image.

6.2.2 Root Node - Statistical analysis

A three-color representation of the multispectral data using channels 60, 17 and 27 (data gathered at wavelengths 0.75μm, 0.46μm and 0.5μm respectively) is shown in Figure 6.3.

The foundation of this study is statistical hyperspectral analysis, as developed and refined over the years by various researchers [56] [87] [35]. Fittingly, the output at this stage is called the Root Node. A potential analysis-route for the stage is

- Assessment of the data for an exhaustive list of scene-classes (Analyst dependent).
- Selection of training data representative of each of the scene-classes in the list compiled above (Analyst dependent).
- Feature selection/extraction from among the 104 spectral channels [87][56] (Computer implementation).
- Classification of data using the method of maximum likelihood (Computer implementation).

In the current study, the Root Node comprised the identification of a comprehensive set of scene-classes {ROOF, ROAD, SHADOW, TREE, GRASS, WATER, PATH}, the selection of (training) data representative of each of the scene-classes, discriminant analysis feature extraction [56], and the construction of a maximum likelihood classifier using the selected training data.

In the selection of training data representative of each of the classes, it was realized that some of the scene-classes (ROOF and ROAD) are a cumulative of several spectrally distinct sub-classes. Consequently, the set of scene-classes was enlarged to {ROOF1, ROOF2, ROOF3, ROOF4, ROOF5, ROOF6, ROOF7, ROOFS, ROAD1, ROAD2, SHADOW, TREE, GRASS, WATER, PATH}. Figure 6.4 is a representation of the associated training data. The results of the classification are shown in Figure 6.5. Note that the sub-classes of ROOF, and those of ROAD, have been merged into their respective groups. Subsequent representations of the output shall also not distinguish among the sub-classes.

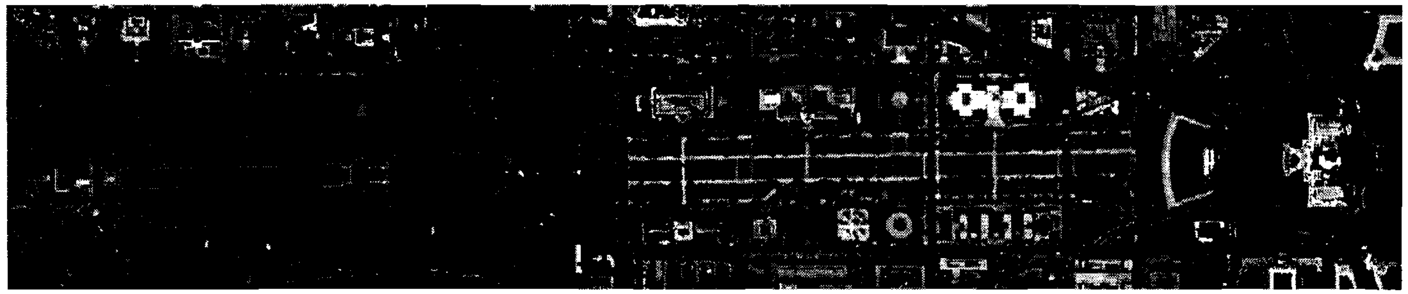


Fig. 6.3: Three color representation of D.C. flightline.

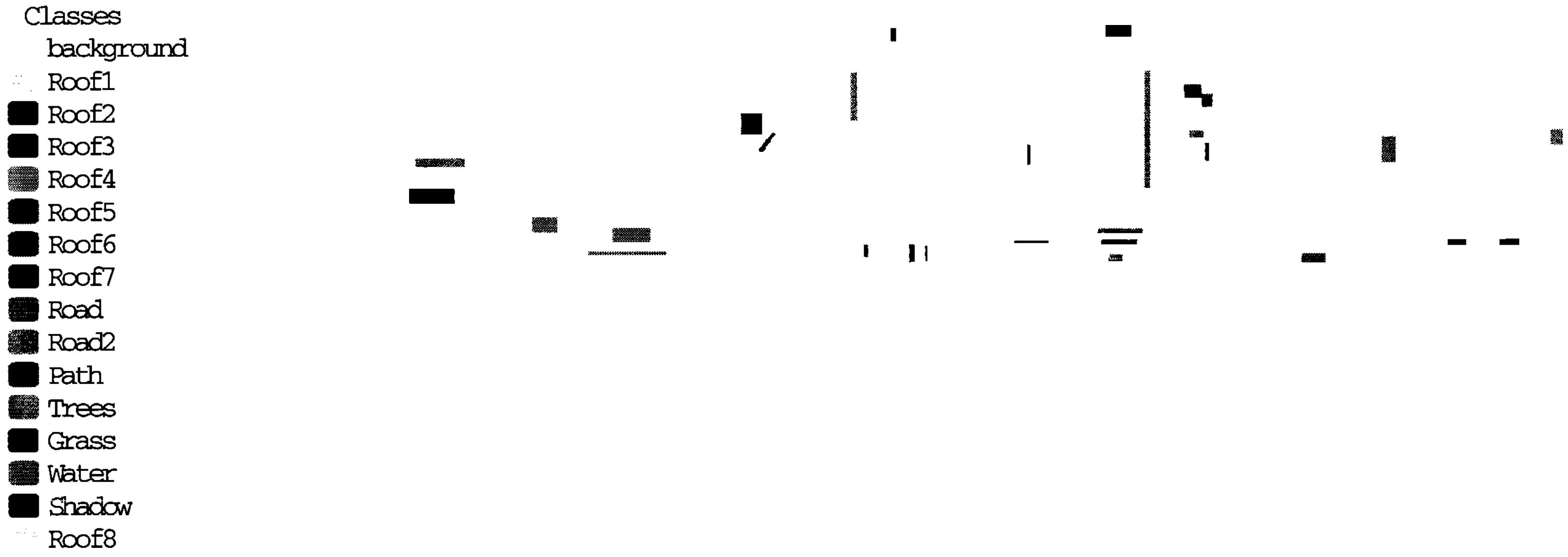


Fig. 6.4: Representation of training fields used in the design of maximum likelihood classifier [Landgrebe3].

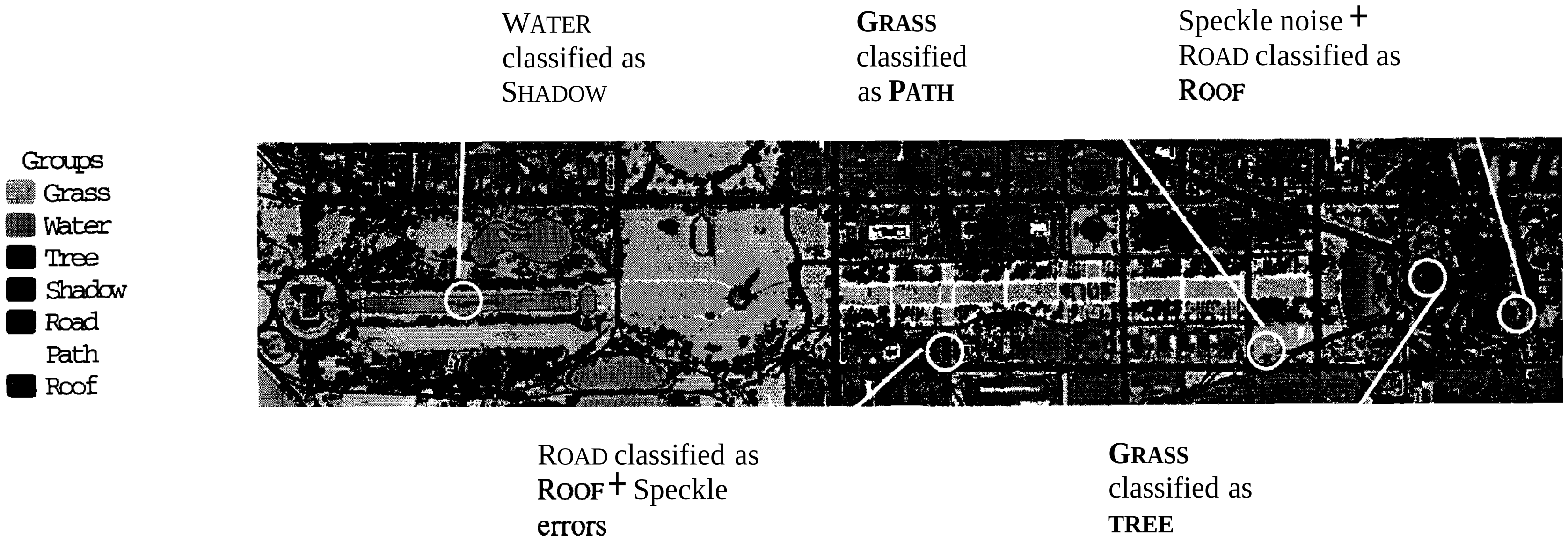

WATER classified as SHADOW
GRASS classified as PATH
Speckle noise + ROAD classified as ROOF
Groups
▨ Grass
■ Water
■ Tree
■ Shadow
■ Road
Path
■ Roof
ROAD classified as ROOF + Speckle errors
GRASS classified as TREE

Fig. *6.5:* Root Node - Scene classification obtained from the use of training data shown in Figure 6.4. Note that the sub-classes for each of ROOF and ROAD have been merged into the respective groups for the representation. Some of the undesirable elements of the output have been flagged above. For a subjective assessment, compare with Figure **6.3.**

The errors in classification are surmised to be a result of spectral similarities among various groups of scene classes, namely -

ROAD, ROOF and PATH;

WATER and SHADOW; and

- TREE and GRASS.

Other undesirable traits of the output include speckle classifications (such as isolated identifications of ROOF on roads, on account of traffic and debris). Since: the multispectral data has been gathered at a high resolution, a visual assessment of the output is a reliable means to guide analysis.

A quantitative assessment of the output is possible, even in the absence of ground 'truth', and is presented in the synopsis of this study, Section 6.2.12.

6.2.3 Node 1 - Separating WATER + SHADOW

From the output at the Root Node, it is evident that spectral separation between WATER and SHADOW can be problematic. It is believed that this is a result of the energy absorption by water, which results in a spectral response that is similar (low in magnitude) to that of a SHADOW region.

The tool used in the separation is the technique proposed in **Part** One, Chapter 4 for segmentation of data, and presented in Section 6.1.2 as the segmentation module using the lattice model.

It is believed that, though there is confusion in separating WATER from SHADOW, the separation of these from other classes is accurate, and complete. Thus, the masking scheme of Section 6.1.5 is applicable.

All data corresponding to either WATER or SHADOW were masked out, and input to the segmentation module detailed in Section 6.1.2. The segmentation was initialized using class-statistics obtained using the training data identified in the top image in Figure 6.6. The output of the segmentation is the lower image in Figure 6.6.

Several experiments with the above scheme were conducted. In each case, the separation of class WATER from class SHADOW had some error. The output shown in Figure 6.5 was accepted as Node 1 of the process, with a resolution to the problem being postponed till a later stage (cf. Section 6.2.9 - Node 8).

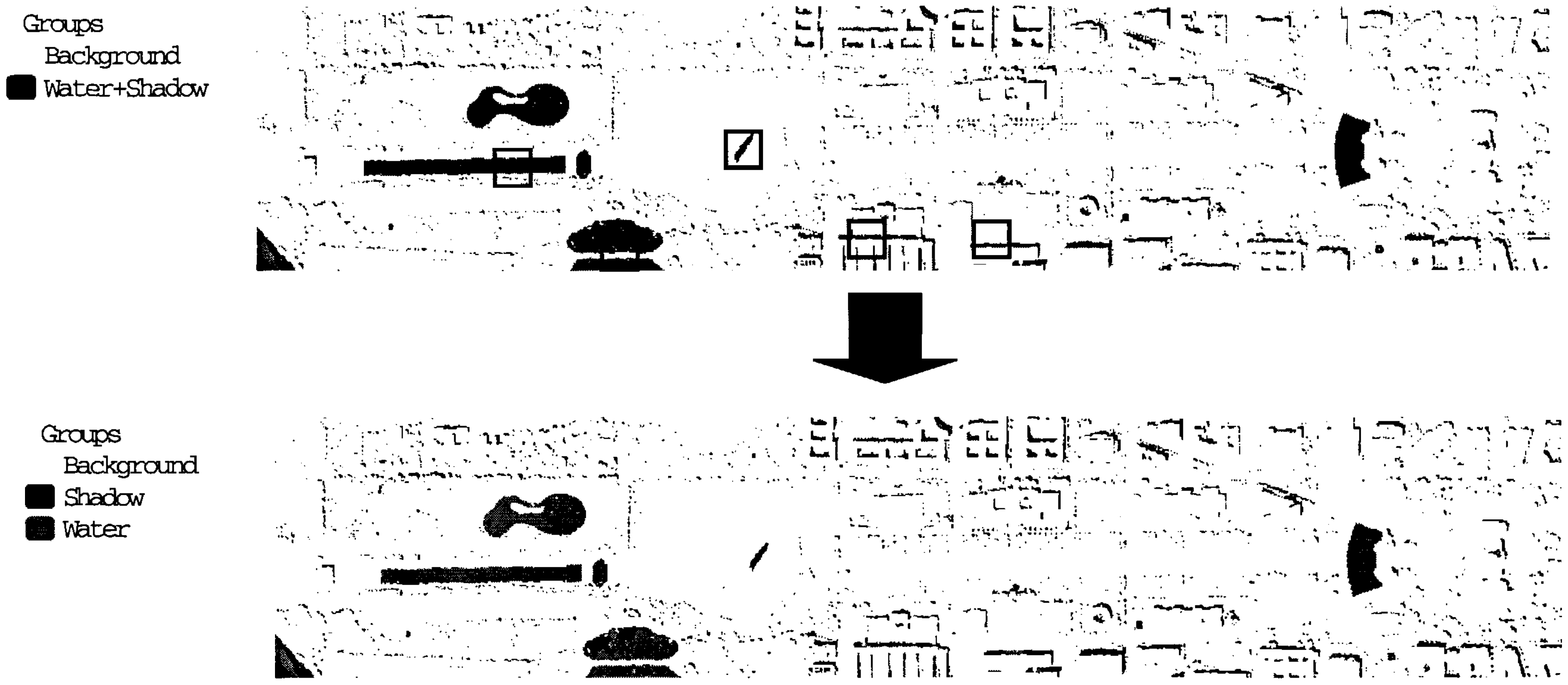


Fig. 6.6: Node 1 - Data corresponding to scene classes SHADOW and WATER were masked out, and segmented using the scheme incorporating the lattice model of Chapter 4. The seed statistics for each of the classes are computed from the training data marked in the top image (enclosed within black boxes for class SHADOW, and within red boxes for class WATER).

The imperfect analysis at Node 1 highlights a flaw in the segmentation algorithm. Recall from Chapter 4, that the lattice model uses neighborhood information to bias decisions. However, since the lattice model is applicable only to binary classification images, the masking scheme is needed (as discussed in Sections 4.3.2 and 6.1.5) to isolate certain sections of the data. These data are presumed to comprise (at most) two scene-classes, and hence the corresponding classification map is binary. A fallout of the masking is that, since the masked spectral data occur as isolated clusters with little interaction, the neighborhood information on the data is depleted and the lattice model is inconsequential. The applicability of any another kind of segmentation scheme is moot, and will be discussed at the next stage, Node 2.

6.2.4 Node 2 - Segmenting SHADOW

As at the previous node, the segmentation module of Section 6.1.2 is used. In this case separation is desired among all data classified as SHADOW at Node 1 into the classes WATER and SHADOW. Figure 6.7 shows all the data identified as SHADOW at Node **1** masked out from the remainder. The data used for initializing the seed statistics of the output classes are also highlighted in Figure 6.7. The output of the algorithm is the lower image in Figure 6.7.

The experiment was repeated several times. In each case, the performance of the algorithm was mediocre, as observed in the output shown in Figure 6.7. This may be expected because, even though SHADOW was originally defined as a scene class, it is distinguished largely through the low magnitude of its spectral response. In general, SHADOW is a composite of sub-classes comprising various low energy responses on diverse materials. The assumption of the unimodal Gaussian distribution for the class may be a poor modeling for this analysis.

Though not essential, this output was retained for this study, and is referred to as Node 2 subsequently.

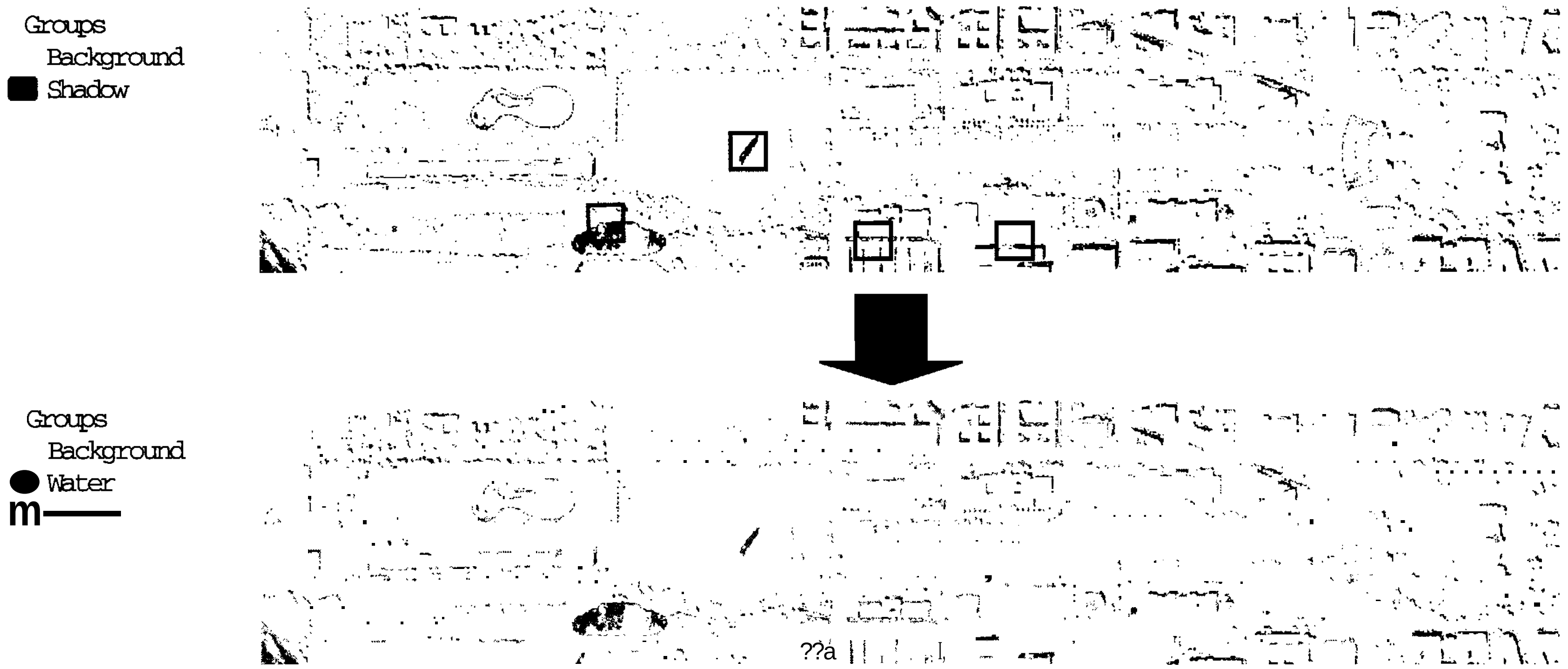


Fig. 6.7: Node 2 - Data corresponding to scene class **SHADOW** from Node 1 in Figure 6.6 were masked out, and segmented using the scheme incorporating the lattice model of Chapter 4. The seed statistics for each of the output classes are initialized using the training data identified in the top image (enclosed within black boxes for class **SHADOW,** and within red boxes for class **WATER).**

6.2.5 Node 3 - Separating GRASS and TREE

From the output at the Root Node, it is evident that though well separable, there are portions of the scene where there is confusion in classifying between GRASS and TREE.

As before, the segmentation scheme of Section 6.1.2 is used. The top image in Figure 6.8a shows the data for classes TREE and GRASS masked out from the remainder, along with the training data used to initialize the seed statistics. The output of the algorithm is the lower image in Figure 6.8a. Note that the separation between GRASS and TREE is cleaner, especially around the far right of the scene. Figure 6.8b highlights the improvement in the classification.

This stage is subsequently referred to as Node 3.

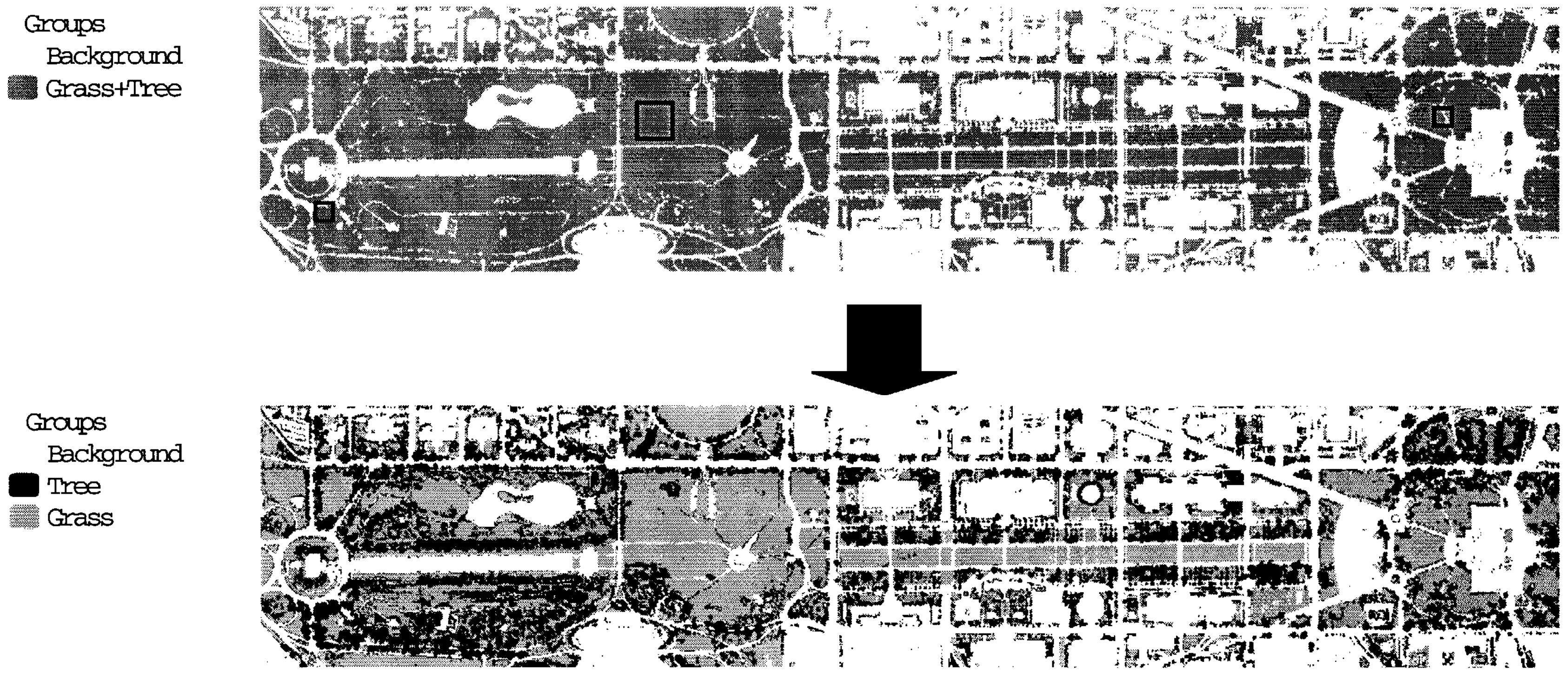


Fig. 6.8a: Node 3 - Data corresponding to scene classes TREE and GRASS were masked out, and segmented using the scheme incorporating the lattice model of Chapter 4. The seed statistics for each of the separated classes are initialized using the training data identified in the top image (enclosed within black boxes for class GRASS, and within red boxes for class TREE).

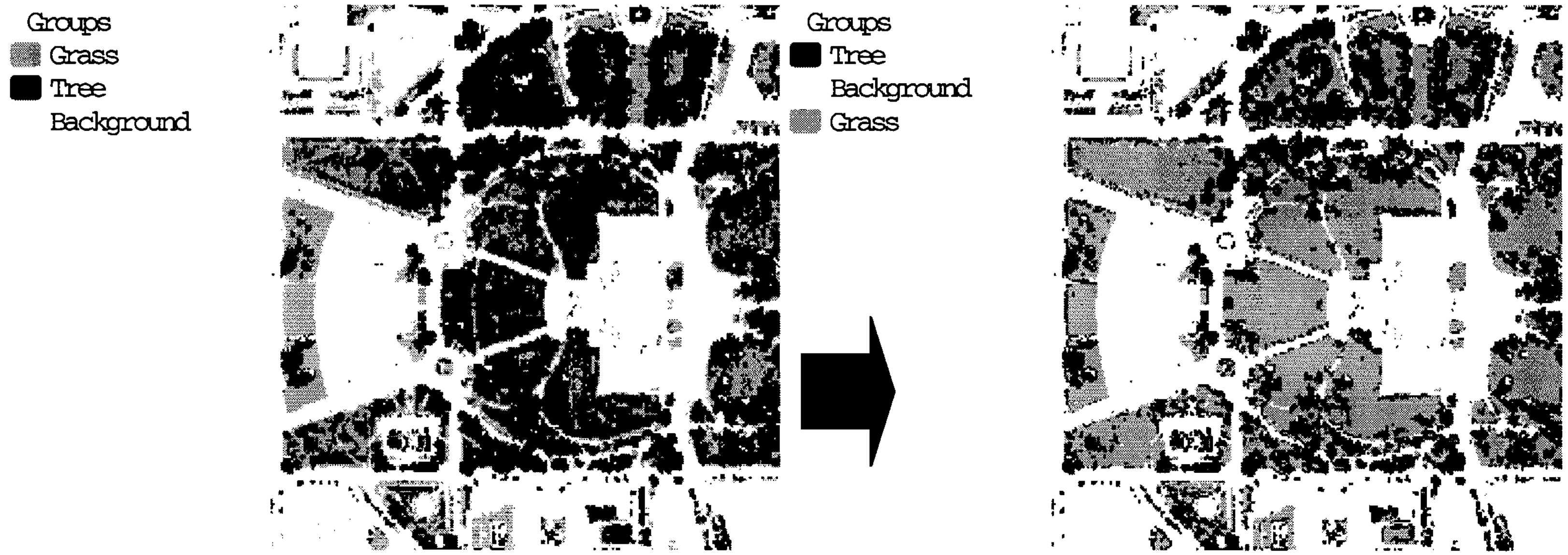


Fig. 6.8b: Sections highlighting the performance of Node 3 in separating GRASS from TREE. Note that the image on the right is extracted from the output at the Root Node (Fig. 6.5) and has all scene-classes other than GRASS or TREE grouped into the background-class.

6.2.6 Node 4 - Extracting rooftops

One of the goals of this analysis is the extraction of roofs in the scene[10]. While the definition of a roof would generally imply "the cover of any building" [91], such a classification is complicated to implement via spectral analysis. There is an immense diversity in the materials used in constructing rooftops, and consequently no single spectral response is representative of the class ROOF. At the Root Node, several spectral sub-classes of ROOF were identified and merged into one group at the end of the analysis. The elements identified as ROOF in Figure 6.5 can be masked from the remainder of the data to yield the image in Figure 6.9. Given multispectral data of the scene, in this way, it is possible to achieve good separation between the composite class ROOF and the other scene-classes. However, this is contingent upon the identification of a comprehensive list of ROOF sub-classes, and a well-executed selection of the respective training data. An alternate means for the analysis is proposed here based on the Digital Elevation Map (DEM) of the scene.

Each pixel in the DEM is representative of the elevation of the corresponding region on the ground. By definition, the information content of this data is directly relevant to the identification of rooftops in the scene. A grayscale representation of the DEM is shown in Figure 6.10. The lighter pixels in the image correspond to elements at higher elevations in the scene.

[10] The discussion of this section has also been published elsewhere [92].

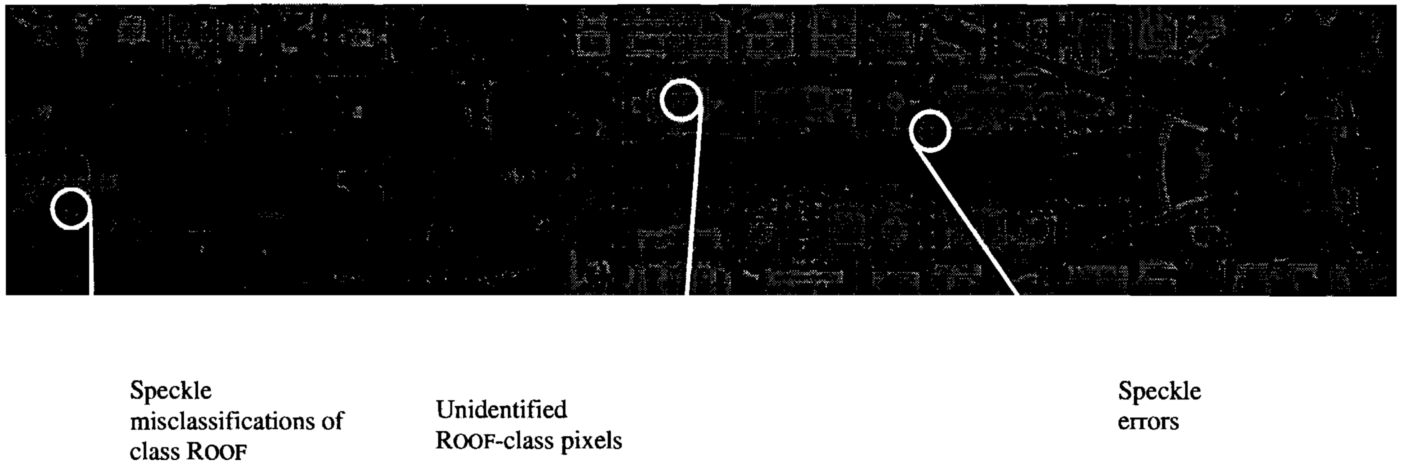

Speckle misclassifications of class ROOF
Unidentified ROOF-class pixels
Speckle errors

Fig. 6.9: Masked out data corresponding to scene-class ROOF with highlighted examples of corruption.

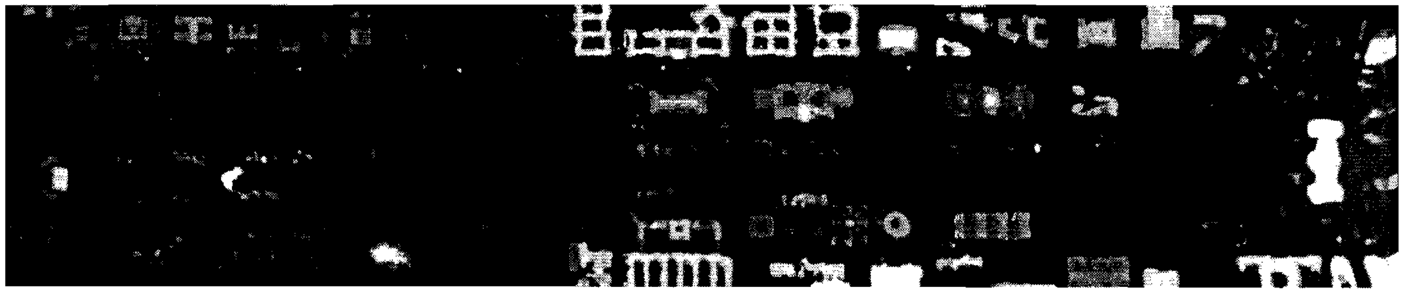


Fig. 6.10: Grayscale representation of Digital Elevation Map (DEM) for D.C. flightline.

The DEM provides information on the rise in elevation of a given area-element in relation to its neighbor. A gradient operator could be used on the data to identify changes in elevation, thereby locating building outlines. A Sobel gradient operator [93] was used on the DEM, and a threshold applied to the result to obtain the image of Figure 6.11.

Figures 6.9, 6.11 are considered representative outputs of the respective techniques for identifying roofs and are compared (cf. Figure 6.12):

- Spectral analysis focuses on pixel-wise identification of the class ROOF. Gradient operator based analysis identifies the building boundaries. In essence, the latter is a scheme to delineate building boundaries, while the other is a pixel classification scheme.
- The output in Figure 6.11 shows extremely thick boundaries to the buildings. While it is possible to thin the delineated scene objects by setting a higher threshold on the output of the gradient operator, this requires extensive analyst manipulation, and is inefficient.
- In general, spectral analysis is more robust over an extended scene. For instance, should the analyst note a different 'type' of building rooftop in isolation, the set of scene-classes can be enlarged and training data included appropriately. On the other hand, analysis of the DEM can be complicated by hilly terrain. In Figure 6.10, note the rise to the Capitol Hill at the right end of the DEM. It is evident that this particular section has to be processed in isolation.
- In Figure 6.9 we can observe considerable speckle misclassifications in the output. In general there is some confusion in separating ROOF - class data from spectrally similar classes ROAD, PATH, and (to some extent) SHADOW.

In highlighting the shortcomings of the respective analyses, it has been implicit that the problems associated with one technique can be alleviated through the use of the other.

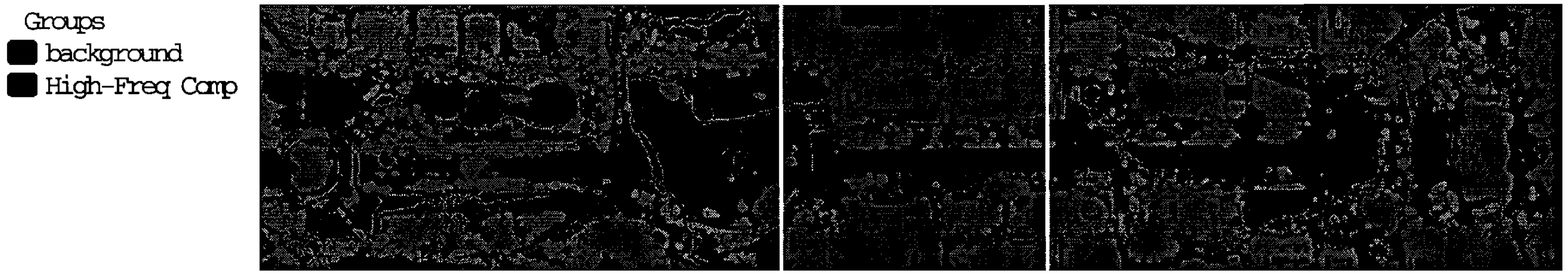


Fig. 6.11: Output of high-pass filtering (Sobel operator) the DEM. The outlined section in the image above is extracted and magnified in Figure 6.12 for a visual comparison with the output of Figure 6.9.

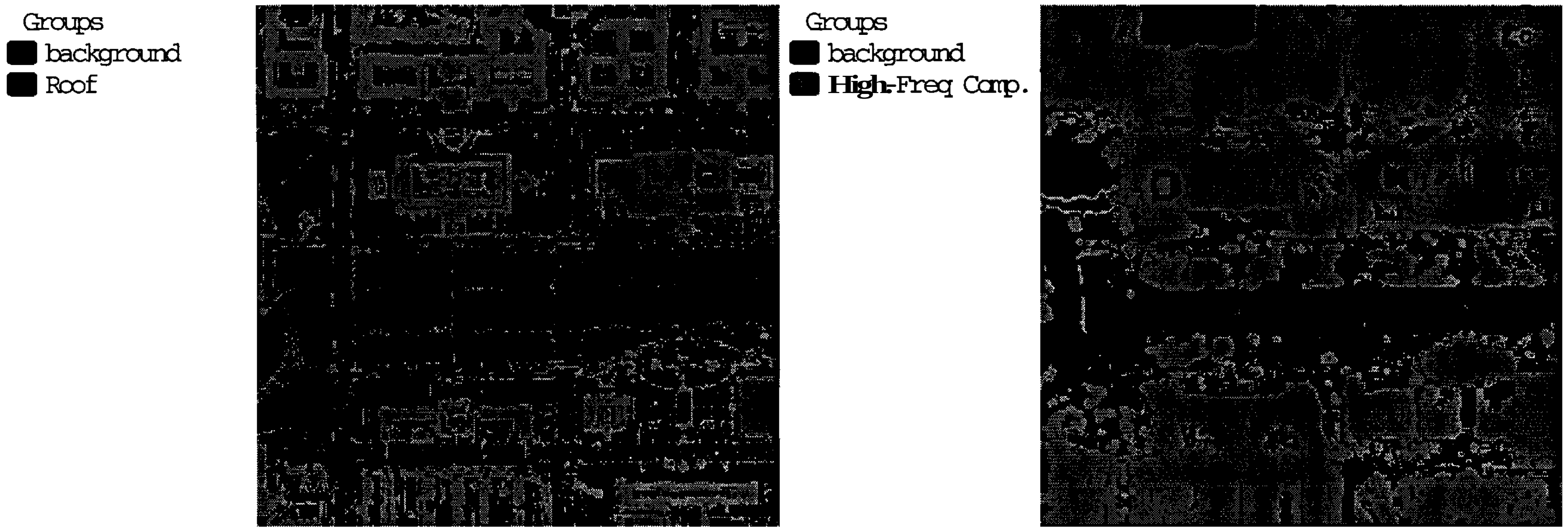


Fig. 6.12: Magnified sections from Figure 6.9 (left image above) and Figure 6.11 (right image above) for illustration of the differences in the techniques generating the respective images.

The proposed solution takes the following route. The output at various stages of the process is displayed in Figures 6.13-15.

- Given the disparity in the two types of the data, concurrent analysis is infeasible. A Class II type fusion, as per Section 5.2, is adopted. It is believed that all the rooftops in the flightline are contained in the classes ROAD, PATH, ROOF and SHADOW identified at the Root Node. The first step in the process is a masking operation on the four classes, excluding the remainder of the data from subsequent analysis.
- Rooftop identification follows via a thresholding operation on the elevations of all data elements identified above. The procedure is designed as a Boolean-type operation in which all data (identified as one of the four classes listed above) below a certain elevation are said to be at ground level; the filtered data are thus identified as building-rooftops.
- Since there is some amount of variation in scene elevation, the elevation threshold has to be locally determined.
- The image is partitioned into several zones, regions of relatively unchanging terrain elevation. The identification of these zones is done manually by visually examining the DEM in Figure 6.10. Zone centroids are selected in the masked data, as shown in Figure 6.13. The respective zones can then be grown outwards from the centroids usng the 'city-block' distance metric[11]. The zone for a given centroid comprises all pixels that are closer to it than any other centroid. The output of the zoning operation for this study is shown in Figure 6.14.
- For each zone, the median elevation for the pixels classified as ROOF, ROAD, SHADOW or PATH is computed. In zones with an insufficient count of rooftop pixels, it is clear that threshold will be biased towards data at ground-elevations. The threshold for a given zone is thus chosen as the average of the median value and the elevation of the zone-centroid. Recall that the zone centroid is manually selected from inspection of Figure 6.3 as being a building/roof pixel.

[11] The city-block distance metric [93] is commonly used in digital image processing applications to measure the distance between two pixels in an image. The distance is the sum of the differences among their respective horizontal and vertical coordinates. For example, the distance between the top right and the bottom left pixels in an image would be the sum of the image's height and width.

- The thresholds, thus computed, were used to get the result shown in Figure 6.15. Note that the identified rooftops are color-coded by their respective zones.

The result of the ROOF identification operation is assimilated with the results of Nodes 1-3 to yield the output displayed in Figure 6.16. Note that data that could not be identified as any in the original set of scene-classes is assigned the class ROOF-RESIDUE.

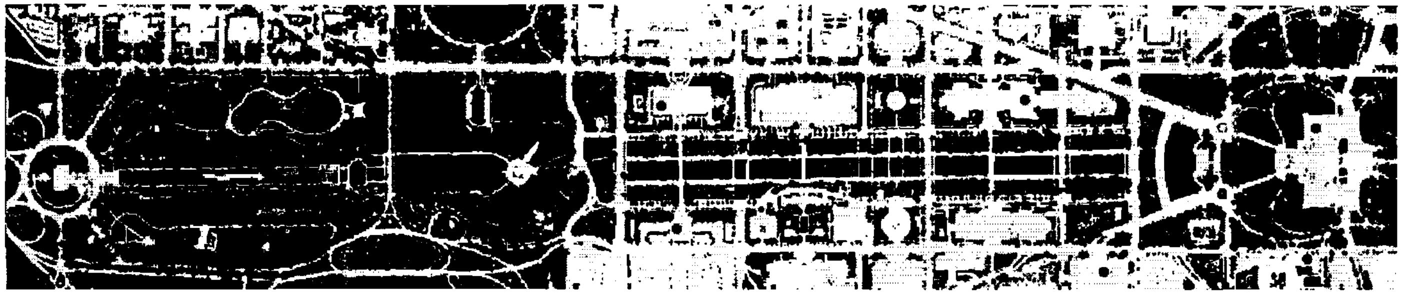


Fig. 6.13: Identification of 'centroids' in data masked out for classes ROOF, PATH, SHADOW and ROAD.

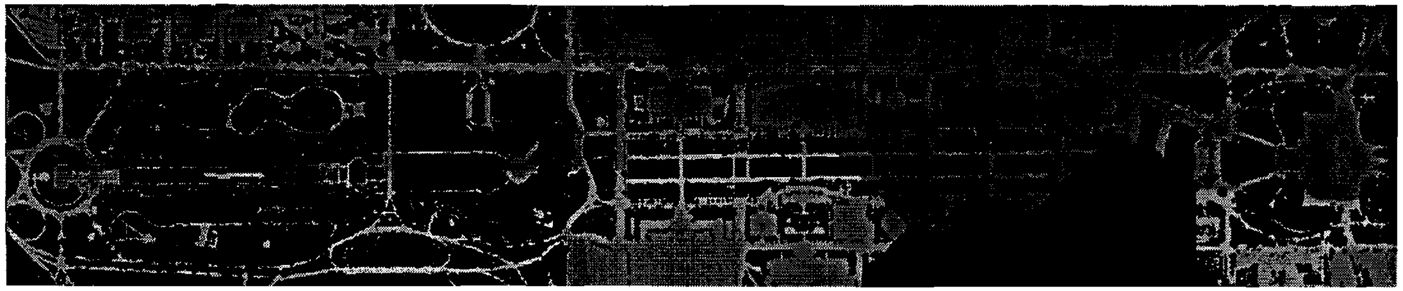


Fig. 6.14: Identification of 'zones' based on centroids shown in Figure 6.13.

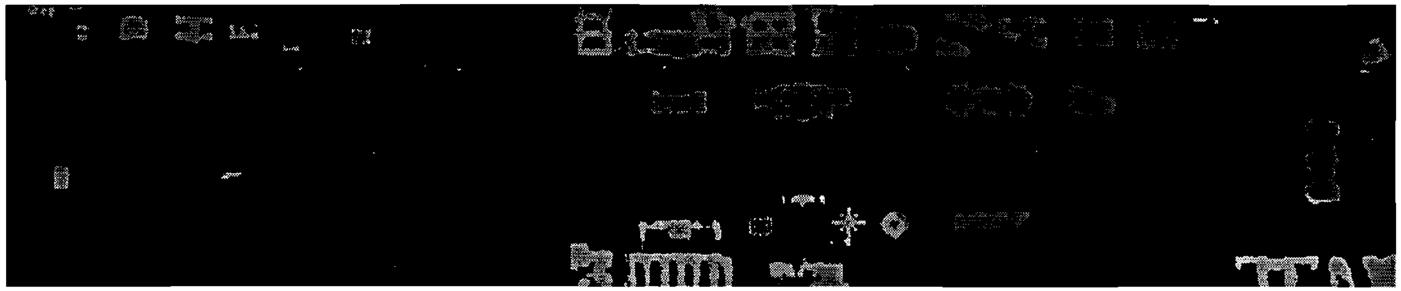


Fig. 6.15: ROOF data identified using elevation thresholds calculated per zone, and applied to all data previously identified as ROOF, PATH, SHADOW or ROAD.

Groups
background
■ Tree
▨ Water
■ Roof
■ Road
Path
■ Roof-residue
▨ Water2
■ Shadow
▨ Grass

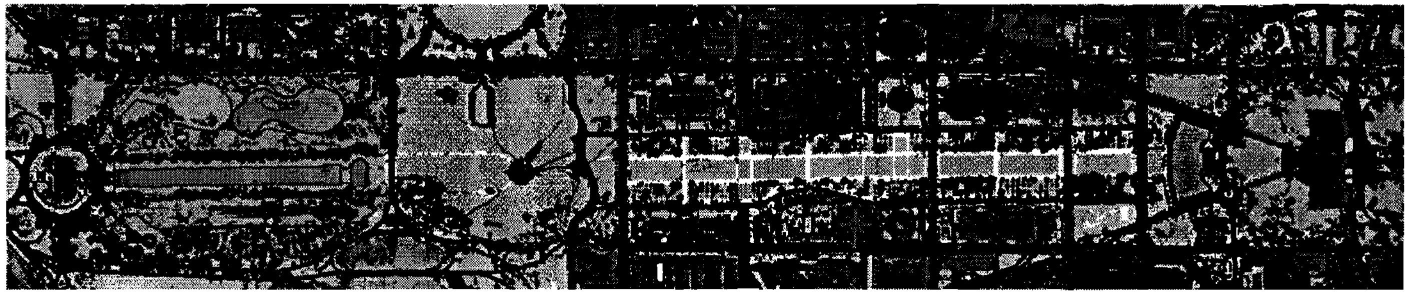


Fig. 6.16: Node 4 - Data corresponding to scene class ROOF were identified using the scheme illustrated in Figures 6.13-15. Segmentation output, as obtained in Nodes 1-3 was incorporated for the above image. Note that all data that could not be identified as among the original set of scene-classes was identified as ROOF-RESIDUE. Also, a new scene class for water has been introduced as per the results of Nodes 1-2, and is labeled WATER2.

6.2.7 Node 5 - Negative training on PATH

Upon close examination of the output at Node 4, it is evident that speckle misclassifications compromise quality of the output.

One way to overcome such errors is to forego the use of the spectral data completely in subsequent processing. The reason for this proposition is that spectral information may not necessarily reflect the target objective. For instance, spectral data on roads is 'corrupted' by presence of gravel, puddles, cars etc. While the statistical spectral classification is true to the data, the returned output is likely cluttered with speckle noise. To counter such problems, the analyst can make the judgment that a given scene-class is isolated to a certain portion of the scene. Figure 6.17 shows the localization of class PATH as a smeared region in the classification map from Node 4, and highlights the aberrations that are sought to be removed from the output. The Negative-Training module discussed previously is employed. The resultant output is displayed in Figure 6.18. It is subsequently referred to as Node 5.

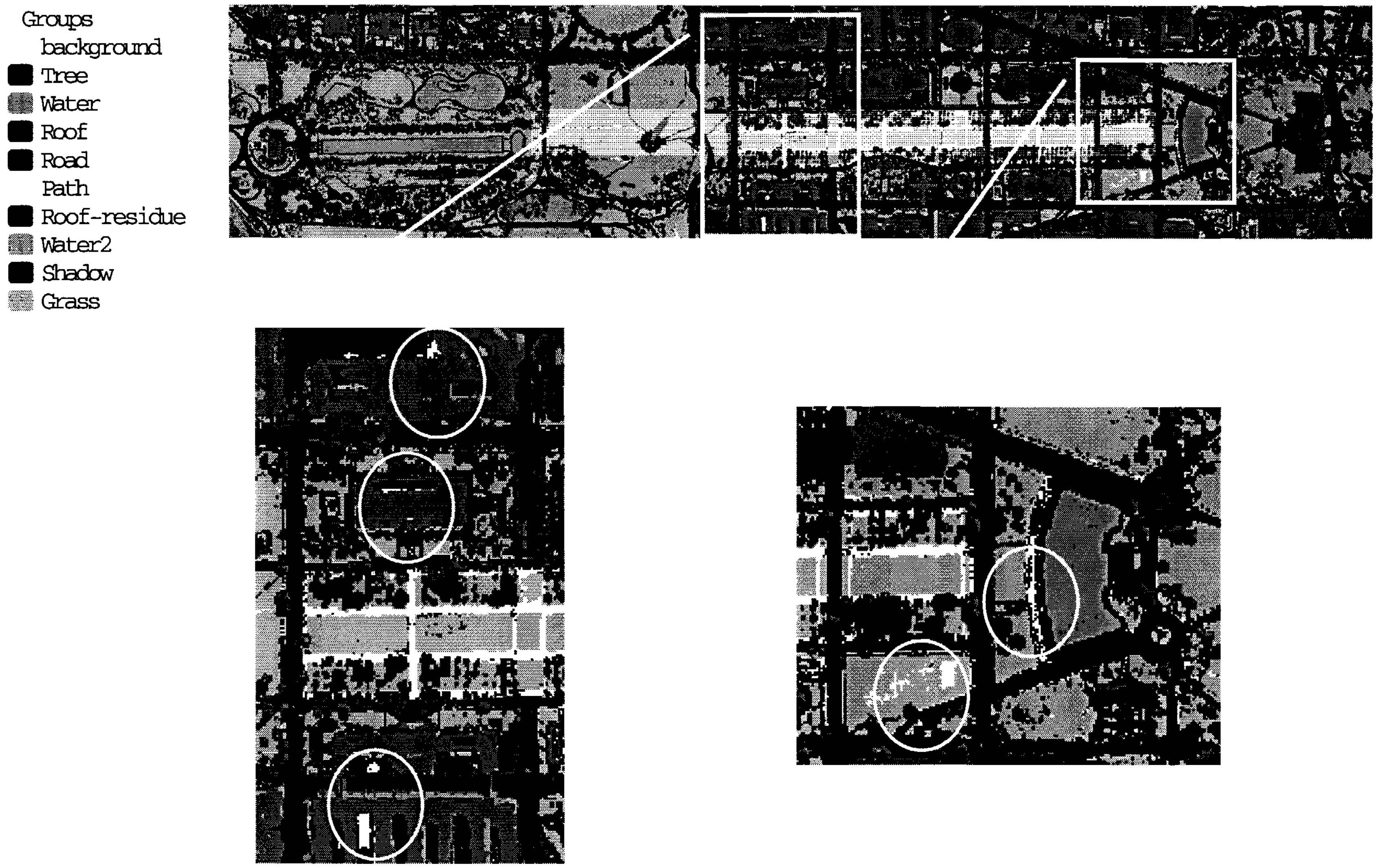


Fig. 6.17: Some of the incorrect identification of class PATH is highlighted above. The smeared window in the top image is believed to contain all the data of class PATH. (See also Figure 6.18.)

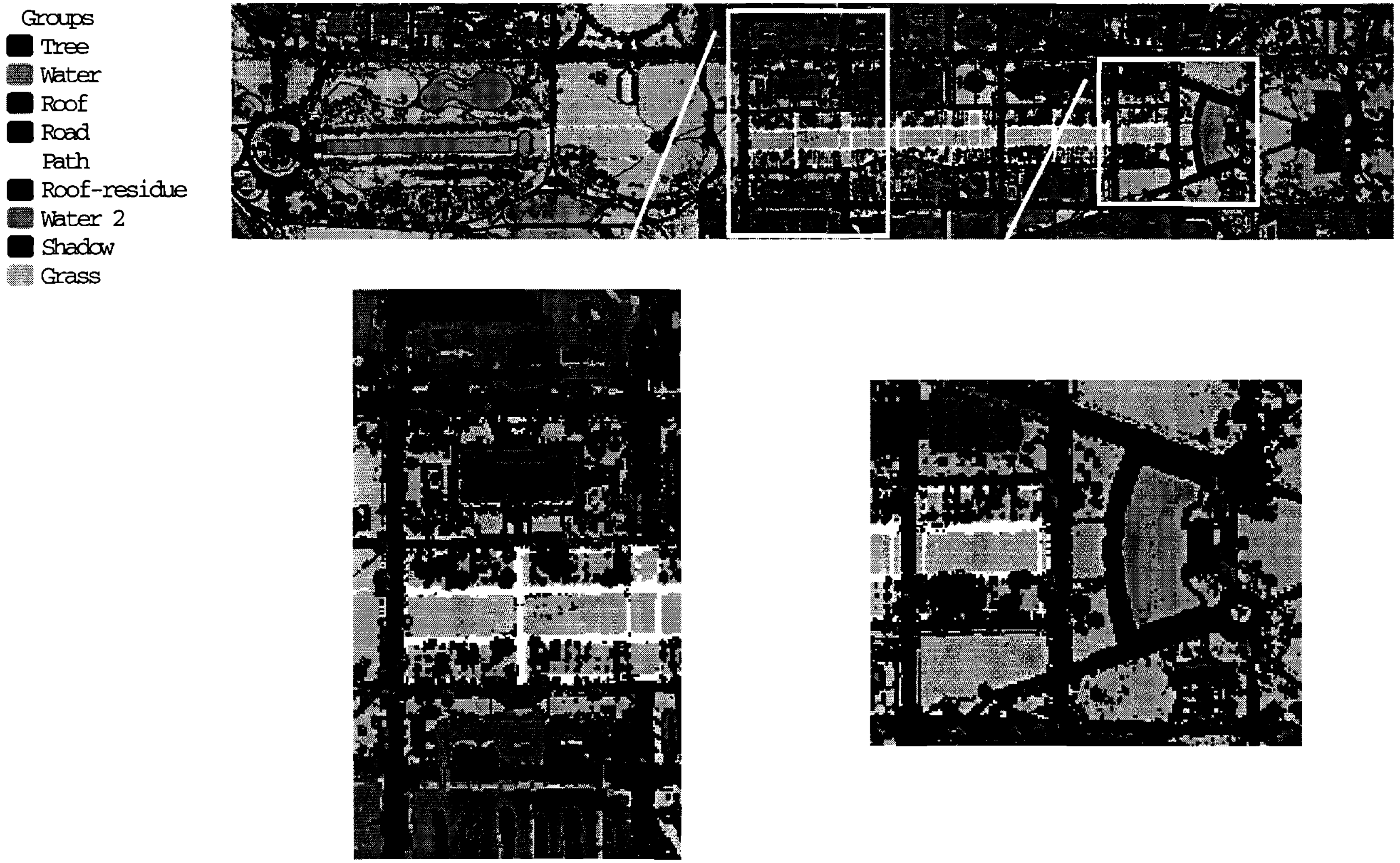


Fig. 6.18: Node 5 - The output of the 'negative training' procedure, as applied to the selected region of Figure 6.17 to clear clutter for class PATH. The re-assigned pixels are labeled to the scene class of most frequency in the respective eight point neighborhoods.

6.2.8 Nodes 6, 7 - Negative training on WATER2, WATER

At this point, it is desired to remove all erroneous classifications of data as Water (or Water2) from the classification map. The Negative Training module is used for the purpose, and the process is illustrated in Figures 6.19 through 6.21.

As in the previous section, it is surmised that the analyst possesses correct information on the locations of water bodies in the scene. As before, such regions containing all WATER2 pixels are identified, as in Figure 6.19, and the Negative-Training module is implemented. The output, Node 6, is displayed in Figure 6.20.

The above process is repeated for the class WATER. (Recall that the development for Node 2 had yielded a second WATER class). The input information and the results are shown in Figures 6.21-22. The output in Figure 6.22 is Node 7 in the process schema.

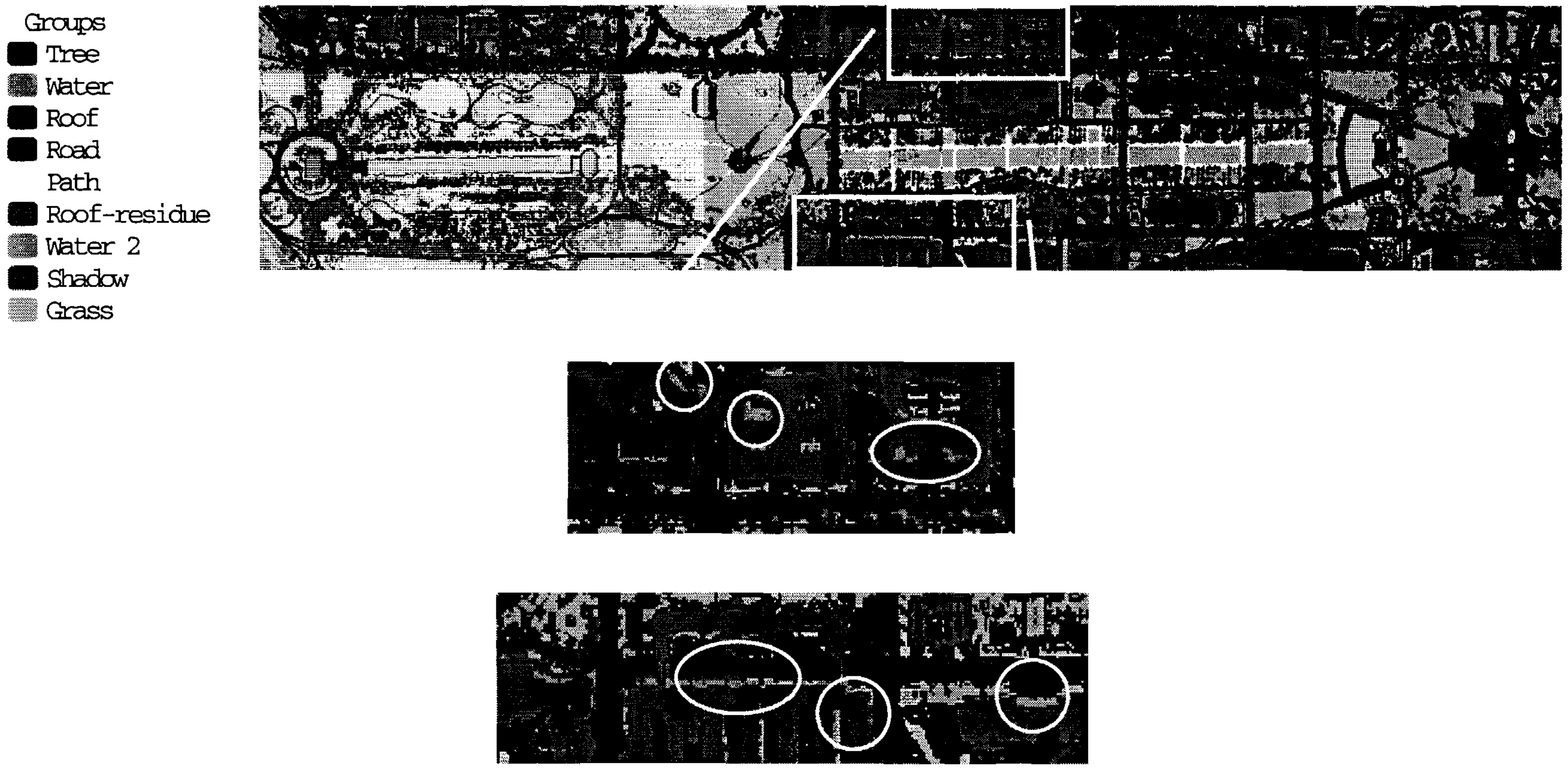


Fig. 6.19: Some of the incorrect identification of class WATER2 is highlighted above. The smeared regions above are believed to contain all the data of class WATER2.

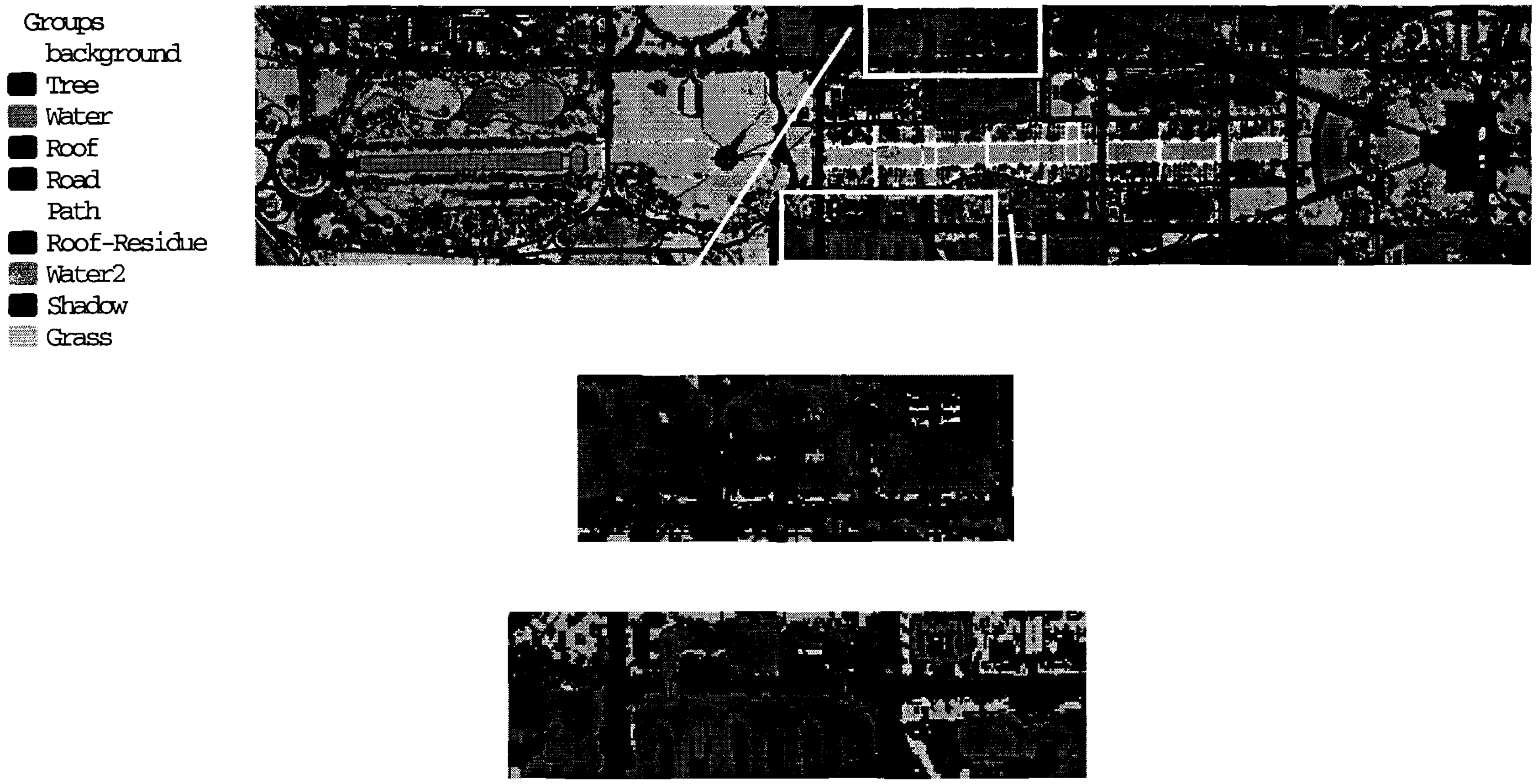


Fig. 6.20: Node 6 - The output of the negative training procedure, as applied to the selected region of Figure 6.19 to clear clutter for class WATER2. The re-assigned pixels are labeled to the scene class of most frequency in the respective eight point neighborhoods.

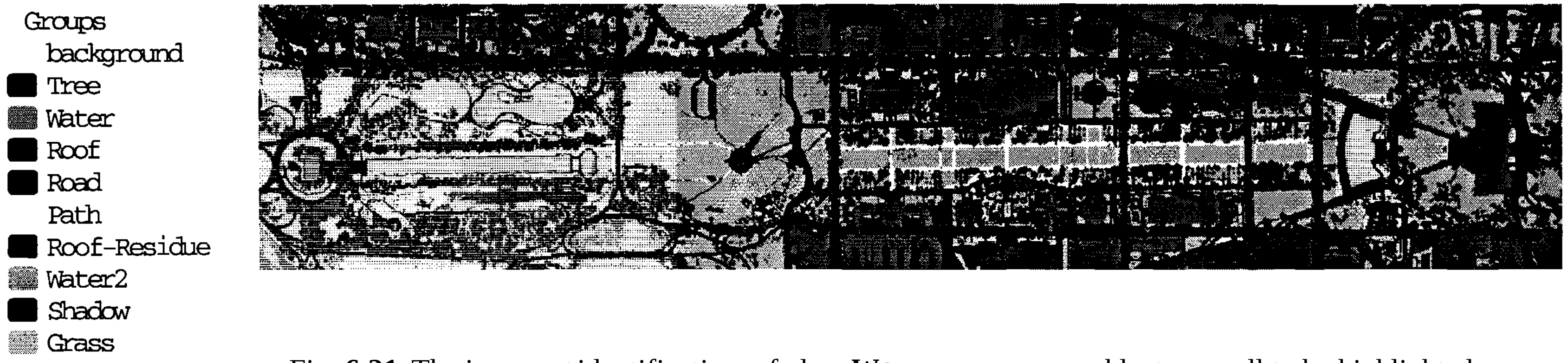


Fig. **6.21:** The incorrect identifications of class **WATER** occur as speckle, too small to be highlighted above. However, the smeared regions above are believed to contain all the data of class **WATER.**

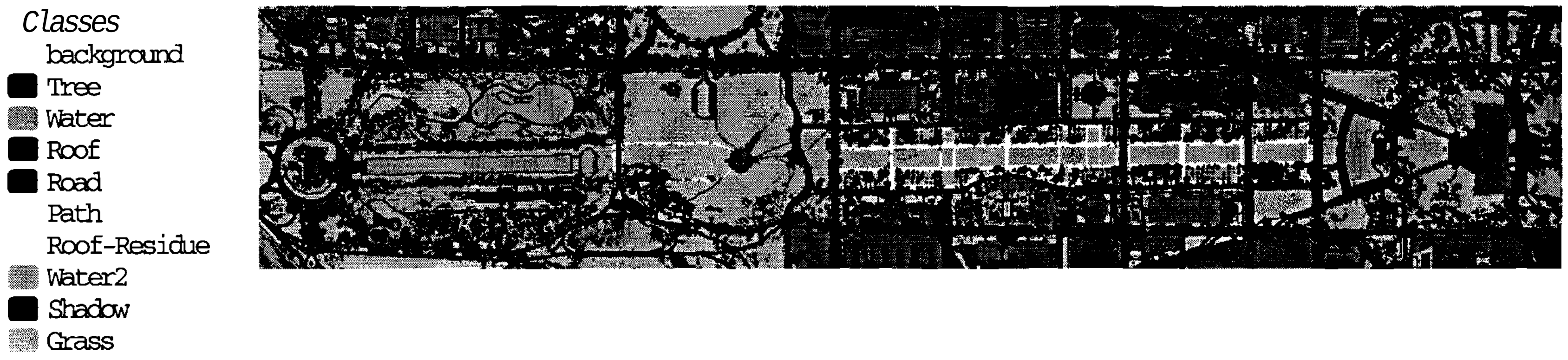


Fig. **6.22:** Node **7** - The output of the negative training procedure, as applied to the selected region of Figure **6.21** to clear clutter for class **WATER.** The re-assigned pixels are labeled to the scene class of most frequency in the respective eight point neighborhoods.

6.2.9 Node 8 - Re-assigning SHADOW (spectral method)

In the discussion following Node 2, it was noted that distinguishing the class SHADOW from the remainder of the data is a difficult task. The original motive for identifying SHADOW in the data was to explain the low energy in the spectral readings over various portions of the image. Failure to do so had resulted in a poor statistical classification of the data. At this stage however, it was concluded that the class SHADOW was non-informative, and had to be removed from the set of scene classes. The maximum likelihood classification module was used to re-assign labels to all data that had been previously classified as SHADOW. The training data for the module were the remainder of the data in the scene, labeled as per the classification map at Node 1. The output of this process, displayed as the lower image in Figure 6.23, is Node 8.

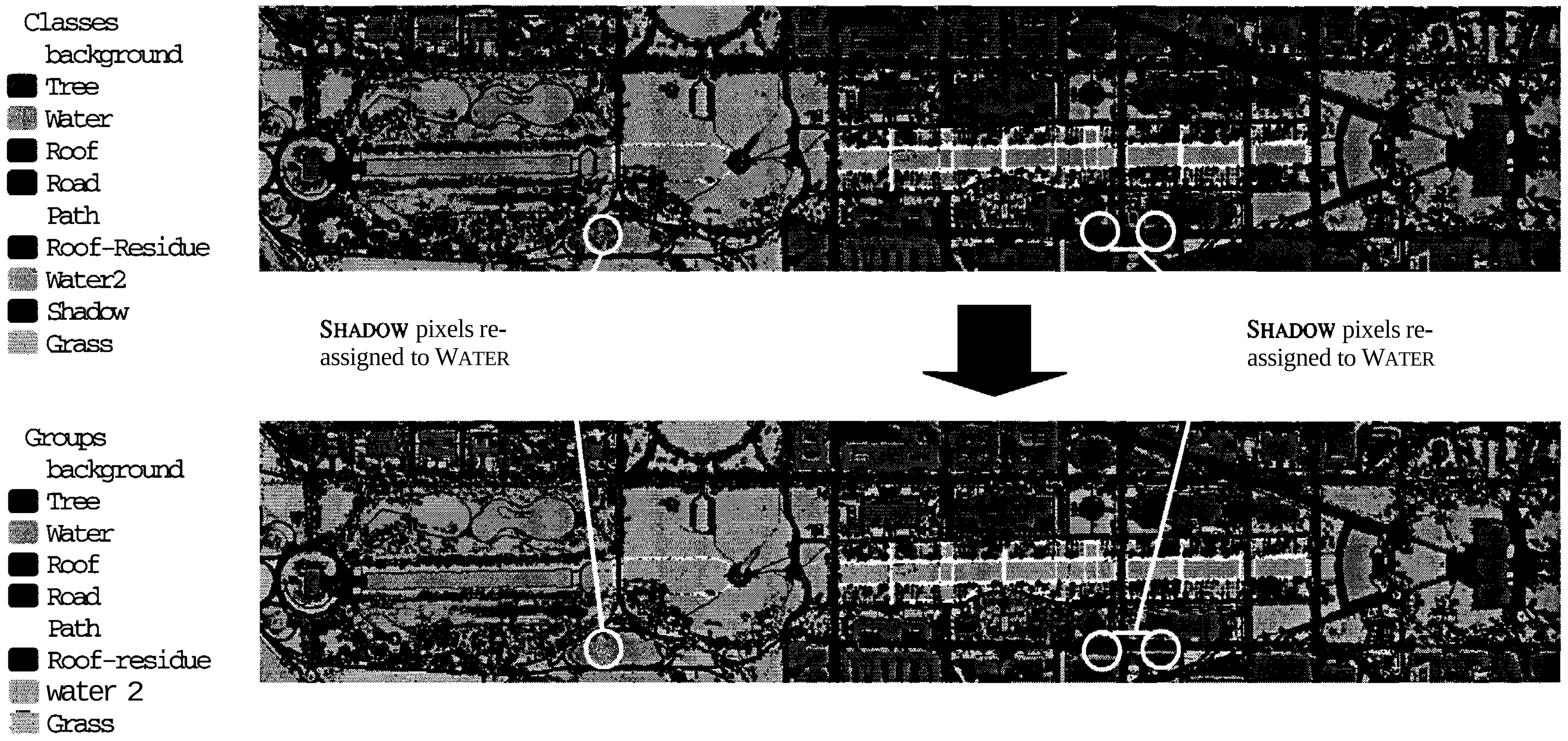


Fig. 6.23: Node 8 - It is concluded that class SHADOW is noise that should be removed. The top image identifies some of the data identified as SHADOW. The lower image shows the output after re-assigning the SHADOW pixels to the spectrally closest scene-class (which is WATER2 for most data).

6.2.10 Node 9 - Negative training on classes WATER + WATER2

As expected, on account of the high spectral similarity between the class SHADOW and WATER2 (or WATER), most of the re-assigned data in Node 8 gets classified as WATER. As previously, these are assumed to be incompatible with the expected output, and removed using the negative training module. The result is Node 9, as shown in Figure *6.25*.

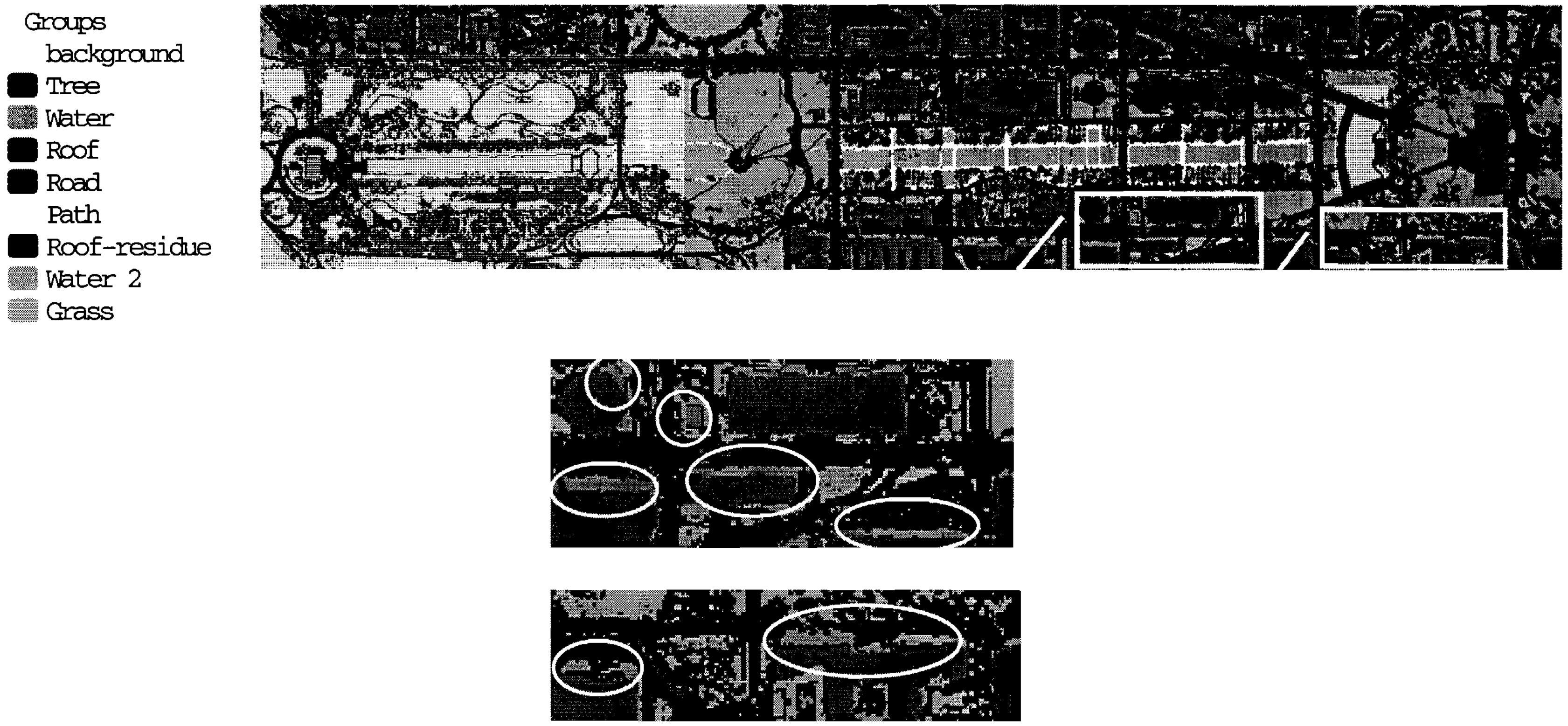


Fig. 6.24: Some of the incorrect identifications of class WATER2 are highlighted above. The smeared regions above are believed to contain all the data of class WATER2.

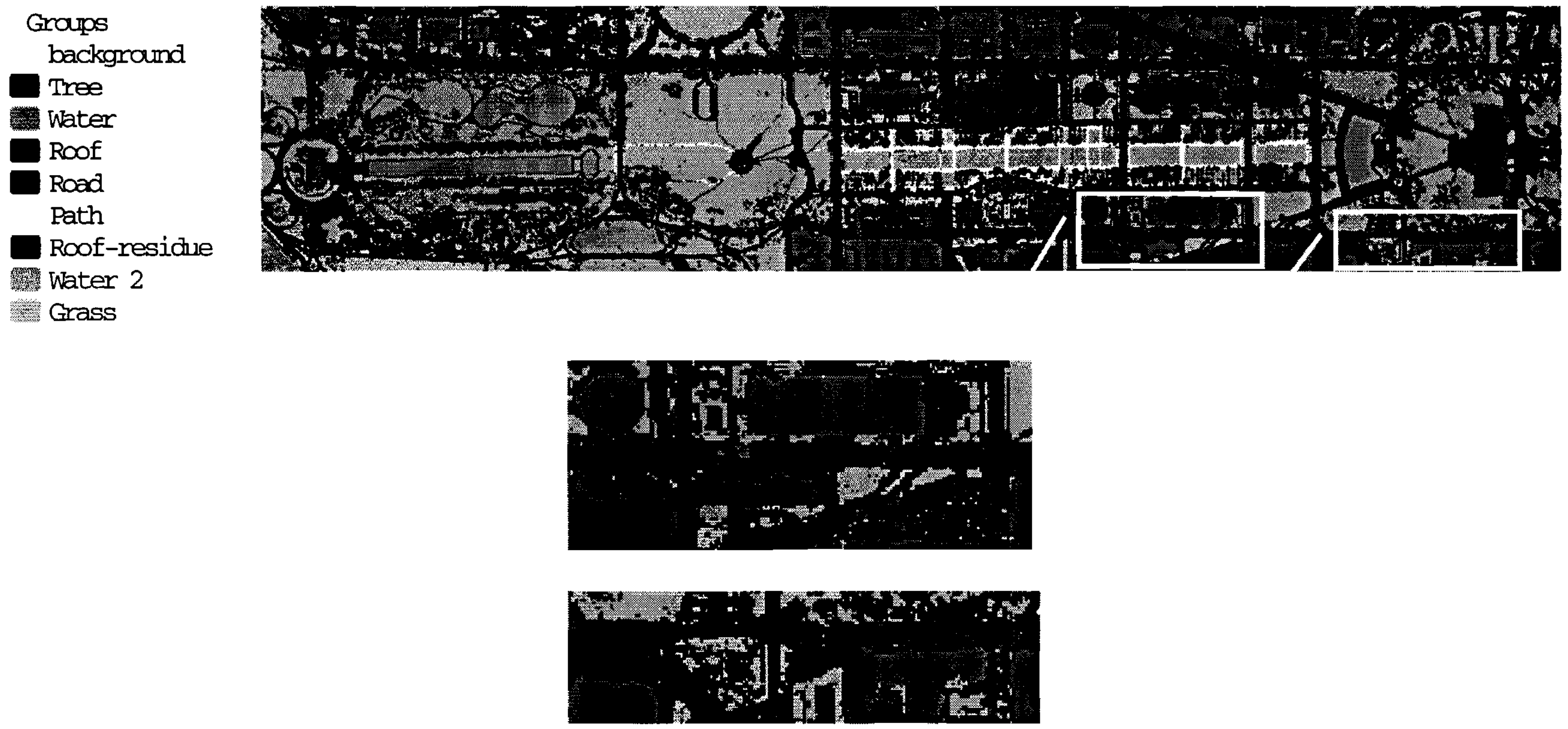


Fig. 6.25: Node 9 - The output of the negative training procedure, as applied to the selected region of Figure 6.24 to clear clutter for class WATER2. The re-assigned pixels are labeled to the scene class of most frequency in the respective eight point neighborhoods.

6.2.11 Node 9b - Reassigning class SHADOW (spatial scheme)

Instead of the scheme used at Nodes 8 and 9, it may be preferred that the re-assignment of the SHADOW pixels is done via a voting scheme on the labels of the respective eight point neighborhoods. This assures that the negative training on WATER, as per Node 8, is not required. The output is shown in Figure ***6.26*** as an alternate to Node 9, and is referred to as Node 9b.

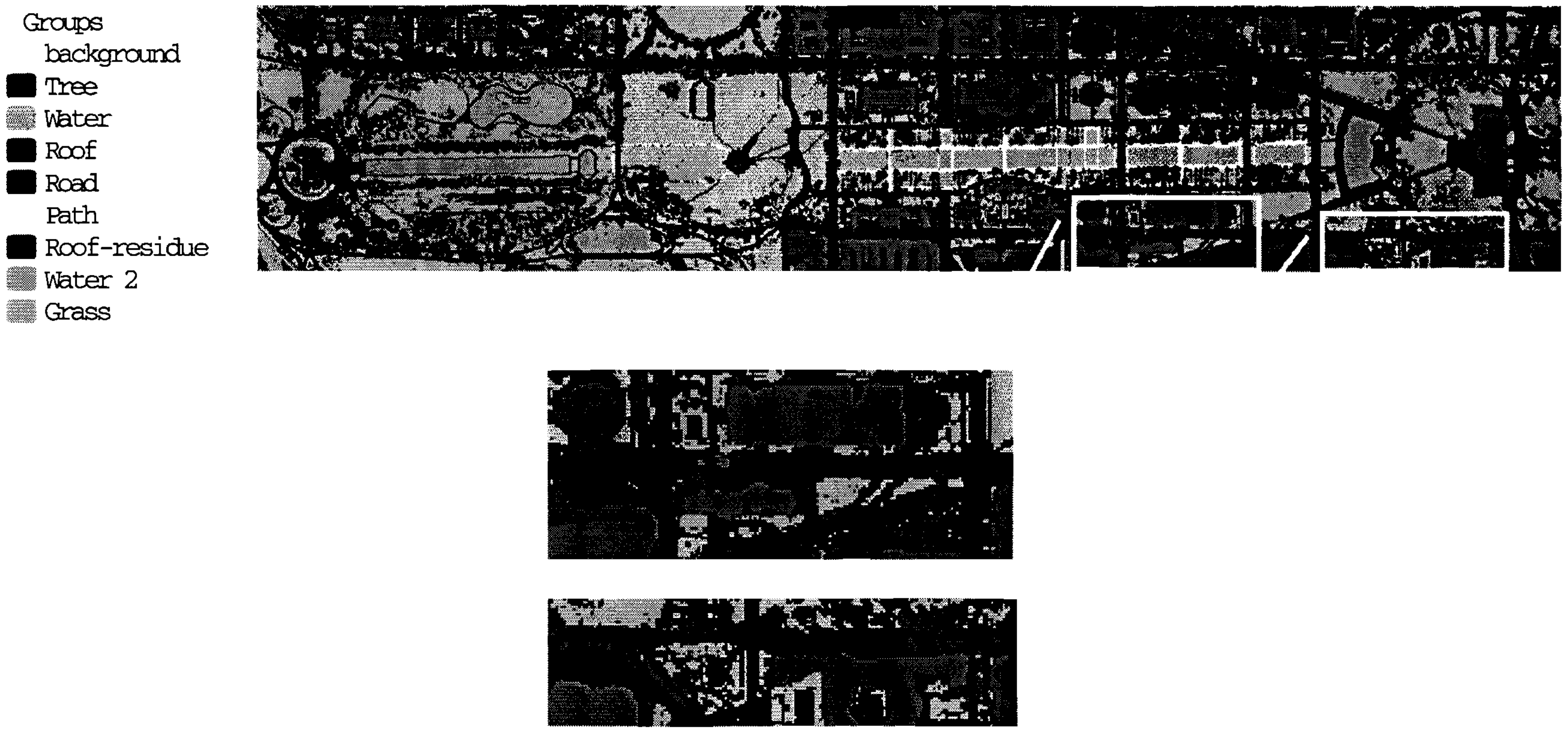


Fig. **6.26:** Node 9b - The output of the negative training procedure, as applied to the Node 7 output of Figure **6.22** to clear clutter for class WATER2. The re-assigned pixels are labeled to the scene class of greatest spectral similarity.

6.2.12 Synopsis

Thus far, the assessments at each level in the process have been subjective. To further justify the development it is believed that a quantitative assessment of the techniques is needed. For this purpose, a set of test data were gathered by a researcher [94] with little knowledge of the analysis presented here. Representative samples of each scene-class were identified in the flightline. The performance of the scene segmentation at various nodes in the analysis can thus be measured as the accuracy of identification of the test data in the respective outputs. Figure **6.27** below is a representation of the identified test data. The classification performance at Node **9** is tabulated in Table **6.1.** For comparison, Table **6.1** also lists the classification performance at the Root Node.

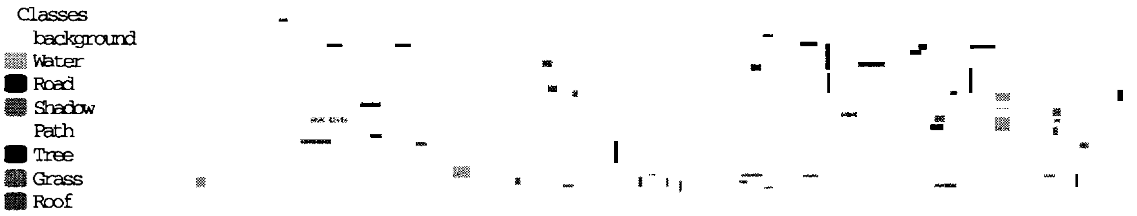


Fig. **6.27:** Representation of test data used in assessing classification performance of D.C. data analysis [94].

Table **6.1**
A quantitative assessment of classification outputs at Root Node and Node **9.**

Scene-class	Number of test samples	Root Node		Node 9	
		Identified number	% accuracy	Identified number	% accuracy
ROAD	**1 056**	**1018**	**96.40**	**1016**	**96.21**
WATER	**1 566**	**1 456**	**92.98**	**1 556**	**99.36**
PATH	**261**	**246**	**94.25**	**238**	**91.19**
TREE	**450**	**429**	**95.33**	**428**	**95.11**
GRASS	**1 378**	**1 029**	**74.67**	**1 270**	**92.16**
ROOF	**1 192**	**974**	**81.71**	**1 059**	88.84[12]
All classes	**5 903**	**5 152**	**87.28**	**5 622**	**94.31**

[12] The scene-class ROOF, has been characterized in the functional sense in Node 9. If scene-class ROOF-RESIDUE is merged with ROOF, the number of pixels in Node 9 correctly identified as 'roof is 1 174, and the overall classification accuracy improves to 97.19%.

The analysis can be condensed into the representation shown in Figure 6.28. It is implicit that the output at a terminal node in the directed graph is retained for the final output without further processing.

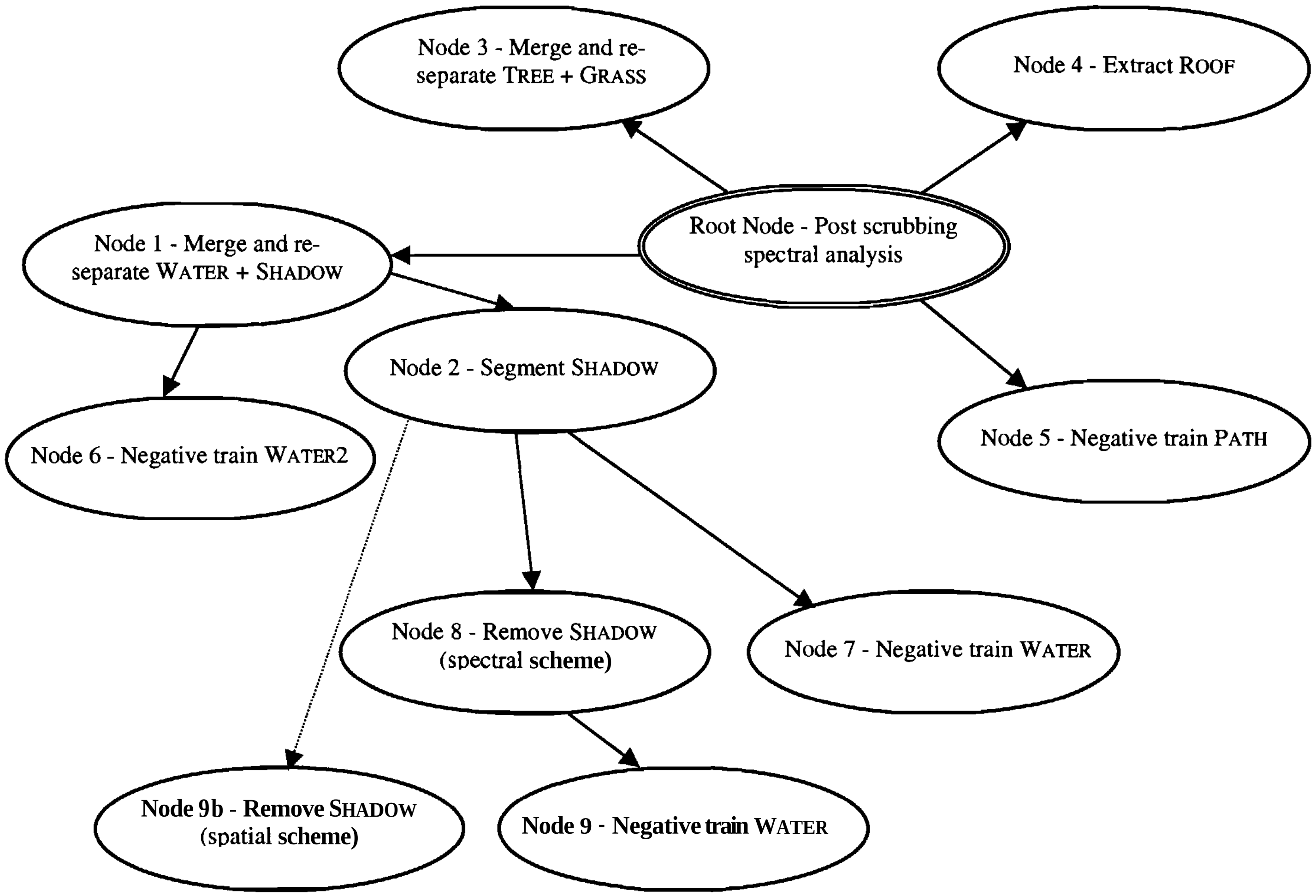

Node 3 - Merge and re-separate TREE + GRASS
Node 4 - Extract ROOF
Root Node - Post scrubbing spectral analysis
Node 1 - Merge and re-separate WATER + SHADOW
Node 2 - Segment SHADOW
Node 6 - Negative train WATER2
Node 5 - Negative train PATH
Node 8 - Remove SHADOW (spectral scheme)
Node 7 - Negative train WATER
Node 9b - Remove SHADOW (spatial scheme)
Node 9 - Negative train WATER

Fig. 6.28: Graph representation of the D.C. data analysis.

Some general conclusions of this study are listed below:

- **A** hierarchical scheme of analysis can be useful, especially if used to isolate problematic data for post-processing independent of the remainder of the data.
- The segmentation scheme using the $2\times N$ model is of limited value if the data to be segmented has insufficient neighborhood information.
- The removal of clutter from the classification output can often be handled without reliance on the spectral data.
- When setting the design goals for the analysis, scene-classes can be decided on the basis of function, or on the basis of composition. For example, in this analysis, the Root Node identified eight distinct materials used in constructing roofs. However, the goal focused on the identification of roofs according to usage, and the final output showed ROOF and ROOF-RESIDLE (the data that were spectrally similar to ROOF, but were functionally distinct).

It should be evident that the procedure followed in this analysis is not unique. It is a moot point whether superior results could be obtained using an alternate scheme. Whatever the chosen design, it is believed that the axioms of data fusion proposed in Chapter 5 and used throughout the development in this Chapter, would be equally applicable.

BIBLIOGRAPHY

[1] J. A. Richards, Remote Sensing Digital Image Analysis, 2nd ed. Berlin: Springer Verlag, 1993.

[2] D. A. Landgrebe, "Information extraction principles and methods for multispectral and hyperspectral image data", in Information Processing for Remote Sensing, C. H. Chen, Ed. New Jersey: World Scientific Publishing Co., 1999.

[3] A. N. Tikhonov, "On the problems with approximately specified information", in Ill-Posed Problems in the Natural Sciences, A. N. Tikhonov. and A.V. Goncharsky, Eds. Moscow: Mir, 1987.

[4] G. E. F. Box and G. Jenkins, Time Series Analysis: Forecasting and Control, Johannesburg: Holden-Day, 1976.

[5] J. Bennett and A. Khotanzad, "Modeling textured images using generalized correlation models", IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 20, no. 12, pp. 1365-1370, December 1998.

[6] R. L. Kashyap and R. Chellappa, "Estimation and choice of neighbors in spatial-interaction models for images", IEEE Transactions on Information Theory, vol. 29, no. 1, pp. 60-72, January 1983.

[7] R. Kindermann and J. L. Snell, Markov Random Fields and their Applications, Providence, RI: American Mathematical Soc., 1980.

[8] N. Ahuja and B. J. Schachter, "Image models", Computing Surveys, vol. 13, no. 4, pp. 374-397, December 1981.

[9] D. Chandler, Introduction to Modem Statistical Mechanics, New York: Oxford University Press, 1987.

[10] C. J. Thompson, Mathematical Statistical Mechanics, New York: Macmillan Co., 1972.

[11] M. Jimbo, T. Miwa, "Algebraic analysis of solvable lattice models", CBMS Regional Conference Series in Mathematics, no. 85, Providence, RI: American Mathematical Soc., 1995.

[12] F. Spitzer, Random Fields and Interacting Particle Systems, Williamstown, MA: Mathematical Assoc. of America, 1971.

[13] A. K. Jain, "Advances in mathematical models for image processing", Proceedings of the ZEEE, vol. 69, no. 5, pp. 122-148, May 1981.

[14] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images", ZEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 6, pp. 721-741, 1984.

[15] G. Winkler, Image Analysis, Random Fields and Dynamic Monte *Carlo* Methods, Berlin: Springer Verlag, 1995.

[16] J. Besag, "Spatial interaction and the statistical analysis of lattice systems" (with discussion), Journal of the Royal Statistical Society, Series B, vol. 36, pp. 192-236, 1974.

[17] J. Besag, "On the statistical analysis of dirty pictures" (with discussion), Journal of the Royal Statistical Society, Series B, vol. 48, pp. 259-302, 1986.

[18] J. Marroquin, S. Mitter and T. Poggio, "Probabilistic solution of ill-posed problems in computational vision", Journal of the American Statistical Association, vol. 82, no. 397, pp. 76-89, March 1987.

[19] P. Dennery, An Introduction to Statistical Mechanics, London: George Allen and Unwin, 1972.

[20] R. C. Dubes, A. K. Jain, S. G. Nadabar and C. C. Chen, "MRF model-based algorithms for image segmentation", in Proceedings - Znternational Conference on Pattern Recognition, pp. 808-814, 1990.

[21] Z. Liang, J. R. MacFall and D. P. Hamngton, "Parameter estimation and tissue segmentation from multispectral MR images", ZEEE Transactions on Medical Imaging, vol. 13, no. 3, pp. 441-449, September 1994.

[22] D. K. Pickard, "Inference for discrete Markov fields: the simplest nontrivial case", Journal of the American Statistical Association, vol. 82, no. 397, pp. 90-94, 1987.

[23] R. J. Elliott, L. Aggoun, "Estimation for hidden Markov random fields", Journal of Statistical Planning & Inference, vol. 50, pp. 343-351, 1996.

[24] N. L. Hjort and H. Omre, "Topics in spatial statistics", Scandinavian Journal of Statistics, vol. 21, pp. 289-357, 1994.

[25] H. Derin and H. Elliott, "Modeling and segmentation of noisy and textured images using Markov random fields", CVGIP: Graphical Models and Image Processing, vol. 54, no. 4, pp. 308-328, July 1992.

[26] M. I. Gurelli and L. Onural, "On a prameter estimation method for Gibbs-Markov random fields", IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 16, no. 4, pp. 424-430, April 1994.

[27] K-C. Ho and B-C. Chieu, "Window-based methods for parameter estimation of Markov random field images", IEICE Transactions on *Information* & Systems, vol. E79-D, no. 10, pp. 1462-1476, October 1996.

[28] S. G. Nadabar and A. K. Jain, "Parameter estimation in Markov random field contextual models using geometric models of objects", IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 18, no. 3, pp. 326-329, March 1996.

[29] J. Moller and R. P. Waagepetersen, "Markov connected component fields", Advances in Applied Probability, vol. 30, no. 1, pp. 1-35, March 1998.

[30] P. Andrey and P. Tarroux, "Unsupervised segmentation of Markov random field modeled textured images using selectionist relaxation", IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 20, no. 3, pp. 252-262, March 1998.

[31] H. Tjelmeland and J. Besag, "Markov random fields with higher-order interactions", Scandinavian Journal of Statistics, vol. 25, pp. 415-433, 1998.

[32] L. Onsager, "Crystal statistics I. A two-dimensional model with an order-disorder transition", Physical Review, vol. 65, pp. 117-149, 1944.

[33] W. Qian and D. M. Titterington, "Multidimensional Markov chain models for image textures", Journal of the Royal Statistical Society, Series B, vol. 53, no. 3, pp. 661-674, 1991.

[34] A. J. Gray, J. W. Kay and D. M. Titterington, "An empirical study of the simulation of various models used for images", IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 16, no. 5, pp. 507-513, May 1994.

[35] D. A. Landgrebe, "The development of a spectral-spatial classifier for earth observational data", Pattern Recognition, vol. 12, pp. 165-175, 1980.

[36] R. Kettig and D. **A.** Landgrebe, "Classification of multispectral image data by extraction and classification of homogeneous objects", IEEE Transactions on Geoscience & Electronics, vol. GE-14, no. 1, pp. 19-26, January 1976.

[37] J. Theiler and G. Gisler, "**A** contiguity-enhanced k-means clustering algorithm for unsupervised multispectral image segmentation", Proceedings of SPIE - the Internaltional Societyfor Optical Engineering, vol. 3159, pp. 108-118, 1997.

[38] B. Jeon and D. A. Landgrebe, "Spatio-temporal contextual classification based on Markov random field model", Proceedings of the International Geoscience & Remote Sensing Symposium (IGARSS), pp. 1819-1822, 1991.

[39] C. **A.** Bouman and M. Shapiro, "Multiscale random field model for Bayesian image segmentation", IEEE Transactions on Image Processing, vol. 3, no. 2, pp. 162-177, March 1994.

[40] M. C. Zhang, R. M. Haralick and J. B. Campbell, "Multispectral image context classification using stochastic relaxation", ZEEE Transactions on Systems, Man & Cybernetics, vol. 20, no. 1, pp. 128-140, January/February 1990.

[41] S. Tadjudin and D. A. Landgrebe, " A decision tree classifier design for high-dimensional data with limited training samples", Proceedings *of the* Znternational Geoscience & Remote Sensing Symposium (ZGARSS), pp. 790-793, May 1996.

[42] M. A. Friedl and C. E. Brodley, "Decision tree classification of land cover from remotely sensed data", Remote Sensing of Environment, vol. 61, no. 3, pp. 399-409, September 1997.

[43] C. Bouman and B. Liu, "Multiple resolution segmentation of textured images", ZEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 13, no. 2, pp. 99-113, February 1991.

[44] R. Hu and M. M. Fahmy, "Texture segmentation based on a hierarchical Markov random field model", Signal Processing, vol. 26, pp. 282-305, 1992.

[45] G. F. Newell and E. W. Montroll, "On the theory of the Ising model of ferromagnetism", Reviews of Modem Physics, vol. 25, no. 2, pp. 353-389, April 1953.

[46] B. Char, G. Gonnet, B. Leong, M. Monagan, S. Watt and K. Geddes, Maple V Library Reference Manual, Berlin: Springer Verlag, 1991.

[47] http://forum.swarthmore.edu/dr.math/faq/faq.cubic.equations.html .

[48] A. D. Sokal, Monte *Carlo* Methods in Statistical Mechanics: Foundations and Algorithms, Lectures at the Cargèse Summer School on "Functional integration: basics and applications", September 1996.

[49] G. S. Fishman, Monte *Carlo:* Concepts, Algorithms and Applications, New York: Springer, 1996.

[50] K. L. Chung, Markov Chains with Stationary Transition Probabilities, Berlin: Springer Verlag, 1960.

[51] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller and E. Teller, "Equations of state calculations by fast computing machines", The Journal of Chemical Physics, vol. 21, pp. 1087-1092, 1953.

[52] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications", Biometrika, vol. 57, no. 1, pp. 97-109, 1970.

[53] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, Numerical Recipes in C: The Art of Scientific Computing, 2nd ed., Cambridge MA: Cambridge University Press, 1992.

[54] USC-SIPI Image Database, http://sipi.usc.edu/services/database/Database.html.

[55] J. Bumgartner, http://www.jbum.com/jbum/pixmagic/planetclouds.jpg.

[56] K. Fukunaga, Introduction to Statistical Pattern Recognition, San Diego CA: Academic Press Inc., 1990.

[57] A. Papoulis, Probability, Random Variables and Stochastic Processes, New York: McGraw-Hill, 1965.

[58] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, "Optimization by simulated annealing", Science, Vol. 220, pp. 671-680, 1983.

[59] L. E. Baum, T. Petrie, G. Soules and N. Weiss, "A maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains", Annals of Mathematical Statistics, vol. 41, no. 1, pp. 164-171, 1970.

[60] E. Redner and H. Walker, "Mixture densities, maximum likelihood and the EM algorithm", *SIAM* Review, vol. 26, no. 2, pp. 195-239, April 1984.

[61] G. H. Ball and D. J Hall, "ISODATA, a novel method of data analysis and pattern classification", International Communications Conference, Philadelphia, June, 1966.

[62] *MATLAB* Release Notes - Version 5.0, Natick MA: The Mathworks Inc., 1995.

[63] D. A. Landgrebe, private communication.

[64] A. Rohatgi and L. Biehl, private communication.

[65] L. Wald, "A European proposal for terms of reference in data fusion",*International* Archives of Photogrammetry & Remote Sensing, vol. XXXII, part 7, pp. 651-654, 1998.

[66] P. S. Chavez, S. C. Sides and J. A. Anderson, "Comparison of three different methods to merge multiresolution and multispectral data: Landsat TM and SPOT panchromatic", Photogrammetric Engineering Remote Sensing, vol. 57, no. 3, pp. 295-303, March 1991.

[67] B. Garguet-Duport, J. Girel, J-M. Chassery and G. Pautou, "The use of multiresolution analysis and wavelets transform for merging SPOT panchromatic and multispectral image data", Photogrammetric Engineering & Remote Sensing, vol. 62, no. 9, pp. 1057-1066, September 1996.

[68] H. Li, B. S. Manjunath and S. K. Mitra, "Multisensor image fusion using the wavelet transform", Graphical Models & Image Processing, vol. 57, no. 3, pp. 235-245, 1995.

[69] K. C. Chou, S. A. Golden and A. S. Willsky, "Multiresolution stochastic models, data fusion, and wavelet transforms", Signal Processing, vol. 34, pp. 257-282, 1993.

[70] K. Dernirbas, "Distributed sensor data fusion with binary decision trees", IEEE Transactions on Aerospace & Electronic Systems, vol. 25, no. 5, pp. 643-649, September 1989.

[71] T. Sawaragi, J. Umemura, O. Katai and S. Iwai, "Fusing multiple data and knowledge sources for signal understanding by genetic algorithm", IEEE Transactions on Industrial Electronics, vol. 43, no. 3., pp. 411-421, June 1996.

[72] N. S. V. Rao, E. M. Oblow, C. W. Glover and G. E. Liepins, "N-Learners problem: fusion of concepts", ZEEE Transactions on Systems, Man & Cybernetics, vol. 24, no. 2, pp. 319-327, February 1994.

[73] I. Bloch, "Information combination operators for data fusion: a comparative review with classification", IEEE Transactions on Systems, Man & Cybernetics, Part A, vol. 26, no. 1, pp. 52-67, January 1996.

[74] C. Dujet and N. Vincent, "Data fusion modeling human behavior", International Journal of Intelligent Systems, vol. 13, no. 1, pp. 27-39, January 1998.

[75] J. L. Crowley and Y. Demazeau, "Principles and techniques for sensor data fusion", Signal Processing, vol. 32, pp. 5-27, 1993.

[76] L. F. Pau, "Behavioral knowledge in sensor/data fusion systems", Journal of Robotic Systems, vol. 7, no. 3, pp. 295-308, June 1990.

[77] N. J. Belkin, P. Kantor, E. A. Fox and J. A. Shaw, "Combining the evidence of multiple query representations for information retrieval", Information Processing & Management, vol. 31, no. 3, pp. 431-448, May-June 1995.

[78] S. D. Kushnier, C. H. Heithecker, J. A. Ballas and D. C. McFarlane, "Situation assessment through collaborative human-computer interaction", Naval Engineers Journal, vol. 108, no. 4, pp. 41-51, July 1996.

[79] D. L. Hall and J. Llinas, "Introduction to multisensor data fusion", Proceedings of the IEEE, vol. 85, no. 1, pp. 6-23, January 1997.

[80] B. V. Dasarathy, "Decision fusion strategies in multisensor environments", IEEE Transactions on Systems, Man & Cybernetics, vol. 21, no. 5, pp. 1140-1154, September-October 1991.

[81] R. T. Antony, "Database support to data fusion automation", Proceedings of the IEEE, vol. 85, no. 1, pp. 39-53, January 1997.

[82] P. K. Varshney, "Multisensor data fusion", Electronics & Communication Engineering Journal, vol. 9, no. 6, pp. 245-253, December 1997.

[83] D. M. McKeown, Jr. , S. D. Cochran, S. J. Ford, J. C. McGlone, J. A. Shufelt and D. A. Yocum, "Fusion of HYDICE hyperspectral data with panchromatic imagery for cartographic feature extraction", IEEE Transactions on Geoscience & Remote Sensing, vol. 37, no. 3, pp. 1261-1277, May 1999.

[84] B. Schniederman, Designing the User Interface: Strategies for *Effective Human-*Computer Interaction, 2nd ed., Reading MA: Addison-Wesley, 1992.

[85] L. Breiman, "Bagging predictors", Machine Learning, vol. 24, no. 2, pp. 123-140, August 1996.

[86] L. Breiman, J. H. Friedman, R. A. Olshen, C. J. Stone, Classification and Regression Trees, Belmont CA: Wadsworth Intl., 1984.

[87] C. Lee and D. A. Landgrebe, "Analyzing high-dimensional multispectral data", IEEE Transactions on Geoscience & Remote Sensing, vol. 31, no. 3, pp. 792-800, July 1993.

[88] C. D. Krivda, "Data-mining dynamite", Byte, vol. 20, no. 10, pp. 97-103, October 1995.

[89] L. Biehl, and David Landgrebe, "MultiSpec - A Tool for Multispectral-Hyperspectral Image Data Analysis", 13th Pecora Symposium, Sioux Falls, SD, August 20-22, 1996.

[90] B. M. Shahshahani and D. A. Landgrebe, "The Effect of Unlabeled Samples in Reducing the Small Sample Size Problem and Mitigating the Hughes Phenomenon," IEEE Transactions on Geoscience & Remote Sensing, vol. 32, no. 5, pp 1087-1095, September 1994.

[91] Webster's Ninth New Collegiate Dictionary, Springfield MA: Merriam-Webster, 1993.

[92] V. Madhok and D. Landgrebe, "Supplementing Hyperspectral Data with Digital Elevation", Proceedings of the International Geoscience & Remote Sensing Symposium (IGARSS), June 28 - July 2, 1999.

[93] A. K. Jain, Fundamentals of Digital Image Processing, Englewood Cliffs NJ: Prentice Hall, 1989.

[94] Y. Zheng, private communication.

[95] S. S. Wilks, Mathematical Statistics, New York NY: Wiley, 1961.

[96] M. B. Priestley, Spectral Analysis and Time Series, London: Academic Press, 1981.

APPENDICES

Appendix A – The Lattice Models

A.1 MATLAB code for solving 2×*N* model

```
function e3 = model2(q, h)
%Input are scalar q, h.
%Output is principal eigenvalue of transfer matrix for 2xN model.
eq=exp(q);
e2q=eq*eq;
e3q=e2q*eq;
e4q=e2q*e2q;
e5q=e2q*e3q;
e6q=e3q*e3q;
e7q=e4q*e3q;
e9q=e4q*e5q;
eh=exp(h);
e2h=eh*eh;
T=[e3q*e2h eh eh 1/eq;
eh eq 1/e3q 1/eh;
eh 1/e3q eq 1/eh;
1/eq 1/eh 1/eh e3q/e2h];
eig(T);
x2=-2*e3q*cosh(2*h)-eq-1/e3q;
A=6*(-5+e4q)*cosh(2*h)+6*sinh(6*q) - 6*cosh(2*q)-6*e6q*cosh(4*h)-
12/e2q;
B=18*cosh(2*h)*(e9q+e5q-eq-(1/e3q))+18*cosh(4*h)*(e7q-
e3q)+18/e5q+72*e3q-90/eq+2*x2*x2*x2;
B=-B;
t=(1/18^(1/3))*(-9*B+(81*B*B+12*A*A*A)^0.5)^(1/3);
e2_bar=A/(3*t)-t;
e3_bar=-e2_bar/2 +0.5*(-4*A-3*e2_bar*e2_bar)^0.5;
e3= (e3_bar-x2)/3;
e2= (e2_bar - x2)/3;
e1=eq-1/e3q;
```

A.2 MATLAB code for solving 3×*N* model

```
function result=model3(q)
a=exp(q);
A=[    5      2      2      1     -1      0      0     -1;
    2      3     -1     -2      0      1     -3      0;
    2     -1      3     -2      0     -3      1      0;
    1     -2     -2      1     -5      0      0     -1;
   -1      0      0     -5      1     -2     -2      1;
    0      1     -3      0     -2      3     -1      2;
    0     -3      1      0     -2     -1      3      2;
   -1      0      0     -1      1      2      2      5];
A=a.^A;
```

```
a2=a*a;
a3=a2*a;
a4=a*a*a*a;
a5=a4*a;
a6=a5*a;
a7=a*a*a*a*a*a*a;
a8=a4*a4;
a10=a5*a5;
a12=a6*a6;
a14=a7*a7;
a16=a8*a8;
a18=a6*a6*a6;
a20=a10*a10;
a22=a10*a12;
a24=a12*a12;
a26=a10*a10*a6;
a28=a14*a14;
a30=a18*a12;
a32=a16*a16;
a34=a18*a16;
a36=a7*a8*a7*a8*a6;
a38=a10*a10*a10*a8;
a40=a10*a10*a10*a10;
e1=(1-a2-a4+a6)/a3;
e2=(-1-a2+a4+a6)/a3;
p1=-12*a16+12*a14+80*a12-120*a10-120*a8+80*a6+12*a4-12*a2+8+8*a18;
q1=12*((3-18*a2+51*a4+12*a8-60*a6+126*a12+84*a10+126*a16+12*a20-
396*a14+51*a24+3*a28-18*a26-60*a22+84*a18)^0.5)*a2;
ot=(1/3)*atan(q1/p1);
if (p1<0)
      ot=pi+ot;
end
x1=((p1^2+q1^2)^(1/6))*cos(ot);
y1=((p1^2+q1^2)^(1/6))*sin(ot);
m1=(a2+1)*(a2-1)/(3*a5);
e3=(x1+1+a2+a4+a6)*m1;
e7=(-(0.5*x1)+a6+a4+a2+1+0.5*y1*(3^0.5))*m1;
e8=(-(0.5*x1)+a6+a4+a2+1-0.5*y1*(3^0.5))*m1;
p2=-36*a22+72*a18-36*a14-36*a10+72*a6-36*a2-96*a16+592*a12-
96*a8+8*a24+48*a20+48*a4+8;
q2=12*((3*a40-24*a2+3+84*a4+6*a8-132*a6-306*a12+336*a10+1287*a16-
2148*a20 -744*a14+1287*a24-306*a28+84*a36+6*a32+336*a30-132*a34-24*a38-
744*a26+564*a22+564*a18)"0.5 )*a2;
ot=(1/3)*atan(q2/p2);
if (p2<0)
      ot=pi+ot;
end
x2=((p2^2+q2^2)^(1/6))*cos(ot);
y2=((p2^2+q2^2)^(1/6))*sin(ot);
m2=(a2+1)/(3*a5);
e4=(x2+1+a4+a4+a8)*m2;
e5=(-(0.5*x2)+a8+2*a4+1+0.5*y2*(3^0.5))*m2;
e6=(-(0.5*x2)+a8+2*a4+1-0.5*y2*(3^0.5))*m2;
result=[e1 e2 e7 e8 e5 e6 e3 e4]
eig(A)'
```

A.3 MATLAB code for generating transfer matrices for L-ary 3×*N* model

```
function A = transfer_D(L)
%L is the number of classes and gives the order of the transfer matrix
ord=L^3;
for p=1:ord,
      for q=1:ord,
            A(q,p)=trans-d((p-1)*ord+q-1, L);
      end
end

function ans = trans_d(a, L)
l=dec2base(a, L, 6);
ans=0;
if (1(1)==l(2)) ans=ans+0.5;
else ans=ans-0.5;
end
if (l(1)==l(3)) ans=ans+0.5;
else ans=ans-0.5;
end
if (l(4)==l(5)) ans=ans+0.5;
else ans=ans-0.5;
end
if (1(4)==l(6)) ans=ans+0.5;
else ans=ans-0.5;
end
if (1(1)==l(4)) ans=ans+1;
else ans=ans-1;
end
if (l(2)==l(5)) ans=ans+1;
else ans=ans-1;
end
if (l(3)==l(6)) ans=ans+1;
else ans=ans-1;
end
```

A.4 MATLAB code for computing magnetization M for 2×*N* model

```
function de3 = m_dh(q, h)
%Input are scalar q, h.
%Output is corresponding magnetization for 2xN model.
eq=exp(q);
e2q=eq*eq;
e3q=e2q*eq;
e4q=e2q*e2q;
e5q=e2q*e3q;
e6q=e3q*e3q;
e7q=e4q*e3q;
e9q=e4q*e5q;
eh=exp(h);
e2h=eh*eh;
T=[e3q*e2h eh eh 1/eq;
eh eq 1/e3q 1/eh;
eh 1/e3q eq 1/eh;
1/eq 1/eh 1/eh e3q/e2h];
eig(T);
x2=-2*e3q*cosh(2*h)-eq-1/e3q;
```

```
A=6*(-5+e4q)*cosh(2*h)+6*sinh(6*q) - 6*cosh(2*q)-6*e6q*cosh(4*h)-
12/e2q;
B=18*cosh(2*h)*(e9q+e5q-eq-(1/e3q))+18*cosh(4*h)*(e7q-
e3q)+18/e5q+72*e3q-90/eq+2*x2*x2*x2;
B=-B;
t=(1/18^(1/3))*(-9*B+(81*B*B+12*A*A*A)^0.5)^(1/3);
e2_bar=A/(3*t)-t;
e3_bar=-e2_bar/2 +0.5*(-4*A-3*e2_bar*e2_bar)^0.5;
e3= (e3_bar-x2)/3;
e2= (e2_bar - x2)/3;
e1=eq-1/e3q;
dAh= 12*(e4q-5)*sinh(2*h)-24*e6q*sinh(4*h);
dx2h= -4*e3q*sinh(2*h);
dBh=36*sinh(2*h)*(e9q+e5q-eq-1/e3q)+72*sinh(4*h)*(e7q-
e3q)+6*x2*x2*dx2h;
dB=-dBh;
dx2=dx2h;
dA=dAh;
dt=(-9*dB+(81*B*dB+18*A*A*dA)/sqrt(81*B*B+12*A*A*A))/(54*t*t);
de2_bar=dA/(3*t) - dt*(1+A/(3*t*t));
de3_bar=-0.5*de2_bar + 0.25*(-4*dA - 6*e2_bar*de2_bar)/sqrt(-4*A-
3*e2_bar*e2_bar);
de3=de3_bar/3 - dx2/3;
de3=de3/model2(q,h);
```

A.5 MATLAB code for computing correlation C for $2\times N$ model

```
function de3 = m_dq(q, h)
%Input are scalar q, h.
%Output is corresponding correlation for 2xN model
eq=exp(q);
e2q=eq*eq;
e3q=e2q*eq;
e4q=e2q*e2q;
e5q=e2q*e3q;
e6q=e3q*e3q;
e7q=e4q*e3q;
e9q=e4q*e5q;
eh=exp(h);
e2h=eh*eh;
T=[e3q*e2h eh eh 1/eq;
eh eq 1/e3q 1/eh;
eh 1/e3q eq 1/eh;
1/eq 1/eh 1/eh e3q/e2h];
eig(T);
x2=-2*e3q*cosh(2*h)-eq-1/e3q;
A=6*(-5+e4q)*cosh(2*h)+6*sinh(6*q) - 6*cosh(2*q)-6*e6q*cosh(4*h)
12/e2q;
B=18*cosh(2*h)*(e9q+e5q-eq-(1/e3q))+18*cosh(4*h)*(e7q-
e3q)+18/e5q+72*e3q-90/eq+2*x2*x2*x2;
B=-B;
t=(1/18^(1/3))*(-9*B+(81*B*B+12*A*A*A)^0.5)^(1/3);
e2_bar=A/(3*t)-t;
e3_bar=-e2_bar/2 +0.5*(-4*A-3*e2_bar*e2_bar)^0.5;
e3= (e3_bar-x2)/3;
e2= (e2_bar - x2)/3;
e1=eq-1/e3q;
```

```
dAq=24*e4q*cosh(2*h)+36*cosh(6*q)-12*sinh(2*q)-36*e6q*cosh(4*h)+24/e2q;
dx2q=-6*e3q*cosh(2*h)-eq+3/e3q;;
dBq=18*cosh(2*h)*(9*e9q+5*e5q-eq+3/e3q)+18*cosh(4*h)*(7*e7q-3*e3q)-
90/e5q+216*e3q+90/eq+6*x2*x2*dx2q;
dB=-dBq;
dA=dAq;
dx2=dx2q;
dt=(-9*dB+(81*B*dB+18*A*A*dA)/sqrt(81*B*B+12*A*A*A))/(54*t*t);
de2_bar=dA/(3*t) - dt*(1+A/(3*t*t));
de3_bar=-0.5*de2_bar + 0.25*(-4*dA - 6*e2_bar*de2_bar)/sqrt(-4*A-
3*e2_bar*e2_bar);
de3=de3_bar/3 - dx2/3;
de3=de3/model2(q, h);
```

A.6 MATLAB code for performance estimation for 2×*N* model

```
%Refer Chapter 4, Section 4.3 for explanation of this subroutine.
% The following are averaged experimental readings on experiments.
table=[ % dim corr   q     mag     h
        % === ====  ====  =====  =====
            2 .765 0.652 -0.078 -0.007;
            3 .822 0.716 -0.064 -0.006;
            4 .855 0.760 -0.059 -0.004;
            6 .905 0.842 -0.038 -0.002;
            8 .935 0.910 -0.024 -0.001;;
            10 .955 0,973 -0.018 -0.001;
            12 .968 1.031 -0.016 -0.001;
            14 .975 1.084 -0.008 -0.000];

plot_length=10000;
d=table(i,1);
corr = table(i,2);
queue= table(i,3);
mag  = table(i,4);
aitch= table(i,5);

num_transitions = round((1-corr)*2*99);
bad-ratio  = 0.5*(num_transitions -1)/(2*99 -1);
good_ratio = 1- bad_ratio;

%Bayesian error
a1=3/8;
a2=3/2;
b=d*log(2);
tot_length=2*plot length+1;
fh1=zeros(tot-length!1);
fh2=zeros(tot_length,1);
ftemp=a1*(2^(0.5*d))*gamma(0.5*d);
gtemp=a2*(2^(0.5*d))*gamma(0.5*d);
for i=1:tot_length,
t=(i-plot_length)/100;
if ((t+b)>=0)
fh1(i)=((t+b)/a1)^(0.5*d-1)*exp(-(t+b)/(2*a1))/ftemp;
fh2(i)=((t+b)/a2)^(0.5*d-1)*exp(-(t+b)/(2*a2))/gtemp;
end
end
e1 = sum(fh1((1+plot_length):tot-length))*.01;
```

```
e2 = 1-sum(fh2((1+plot_length):tot-length))*.01;
[el e2 0.5*(el+e2)1
hold off
plot(0.01*((1:tot_length)-plot_length), fhl, 'r.',
0.01*((1:tot_length)-plot_length), fh2, 'b.')
hold on
bl=b-2*queue-aitch;
b2=b+2*queue-aitch;%1.178;
fhl_A=zeros(tot-length,1);
fh2_A=zeros(tot-length,1);
fhl_B=zeros(tot-length,1);
fh2_B=zeros(tot length,1);
for i=l:tot_length,
t=(i-plot_length)/100;
if ((t+bl)>=0)
fhl_A(i)=((t+bl)/al)^(0.5*d-1)*exp(-(t+bl)/(2*al))/ftemp;
fh2 A(i)=((t+bl)/a2)^(0.5*d-1)*exp(-(t+bl)/(2*a2))/gtemp;
end-
if ((t+b2)>=0)
fhl_B(i)=((t+b2)/al)^(0.5*d-1)*exp(-(t+b2)/(2*al))/ftemp;
fh2_B(i)=((t+b2)/a2)^(0.5*d-1)*exp(-(t+b2)/(2*a2))/gtemp;
end
end

% I plot(1:plot_length, fhl_A,'m-', 1:plot_length, fh2_B, 'g-')
el-A = sum(fhl_A((1+plot_length):tot_length))*.01;
e2_B = 1-sum(fh2_B((1+plot_length):tot_length))*.01;
plot(0.01*((1:tot_length)-plot_length), fhl_B,'m-',
0.01*((1:tot_length)-plot_length), fh2_A, 'g-')
el-B = max(sum(fhl_B((1+plot_length):tot_length))*.01, 0.0);
e2_A = max((1-sum(fh2_A((1+plot_length):tot_length))*.01), 0.0);
[el_B e2_A 0.5*(el_B+e2_A)];
ftemp = 0.98*el_B +0.02 * el;
gtemp = 0.98*e2_A +0.02 * e2;
[ftemp gtemp 0.5*ftemp+0.5*gtemp]
ftemp = 0.98*(good_ratio*el_B + bad-ratio * el_A) +0.02 * el;
gtemp = 0.98*(good_ratio*e2_A + bad-ratio * e2_B) +0.02 * e2;
[ftemp gtemp 0.5*(ftemp+gtemp)]
```

A.7 Correlation calculation for test images

For each of the sample images below, it is observed that pixel-label transitions occur in a regular fashion. It suffices to examine a sample 2×N chain that contains the representative transition to measure the correlation for the image. The applicable formula is

$$\text{correlation} = 1 - \frac{\text{number of transitions}}{\text{number of total pixels in length of } 2\times \text{N chain}}$$

For example, the representative chain for the image in Figure **A.l** would contain 1 transition in a chain containing 2.99 pixels. Thus the correlation would be 0.995. The q is obtained from Table 3.1 (note that *h*=0). Note that the above formula can also be used to deduce the number of transitions in a given image, from the reading of the image correlation.

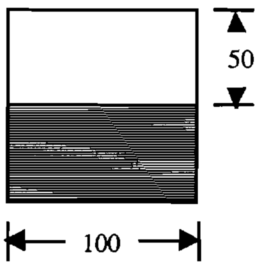


Fig. A.1: For given 100×100 image, correlation = 0.995, q≈1.44.

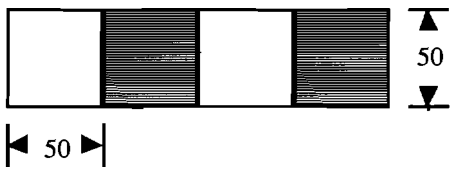


Fig. A.2: For given 200x50 image, correlation= 0.992, $q\approx$ 1.35.

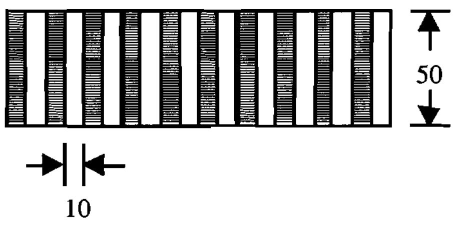


Fig. A.3: For given 200x50 image, correlation= 0.9523, $q\approx 0.987$.

A.8 Model characteristics and performance of estimators

Recall the terminology of Chapter 3:

- M is an unbiased estimator of the magnetization M;
- $\hat{C}$ is an unbiased estimator of the correlation C;
- $P_X(X)$ is the distribution of the system state X and from Eauation 3.1

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

- Z is a normalizing factor, also known as the partition function calculated as
$$Z = \sum_{\{X\}} e^{-\mathrm{E}(X)};$$
- N is the number of linked elements in the lattice chain X;
- E(•) is the energy function, and is a measure on the space of system configurations **X.** For the $2{\times}N$ model for $X{=}\mu\mu'$ (see Figure **3.7** for details) the energy function is defined as
$$E(X) = -q\sum_{i=1}^{N}\left(\mu_i\mu_{i+1} + \mu_i'\,\mu_{i+1}' + \mu_i\mu_i'\right) - h\sum_{i=1}^{N}\left(\mu_i + \mu_i'\right).$$

Thus

$$\frac{\partial Z}{\partial q} = \sum_{\{X\}} -\frac{\partial \mathrm{E}(X)}{\partial q} e^{-\mathrm{E}(X)}$$
$$= \sum_{\{X\}}\left[\sum_{i=1}^{N}\left(\mu_i\mu_{i+1} + \mu_i'\,\mu_{i+1}' + \mu_i\mu_i'\right)\right] e^{-\mathrm{E}(X)}$$

and

$$\frac{1}{Z}\frac{\partial Z}{\partial q} = \sum_{\{X\}}\sum_{i=1}^{N}(\mu_i\mu_{i+1} + \mu_i'\,\mu_{i+1}' + \mu_i\mu_i')P_X(X). \tag{A.1}$$

The above expression may be interpreted as the expectation (with respect to the distribution P_X) of the pair-wise correlation C in the lattice. Similarly, the expected value of the lattice magnetization M is obtained as

$$\frac{1}{Z}\frac{\partial Z}{\partial h} = \sum_{\{X\}}\sum_{i=1}^{N}(\mu_i + \mu_i')P_X(X). \tag{A.2}$$

The remainder of this section illustrates some properties of the $2{\times}N$ lattice model. Figure A.4 plots the lattice correlation and magnetization over a range of q and h. Note that lattice magnetization sees a sharp transition about h=0, for large values of q. However, this thesis is not concerned with the usage of the proposed model to explain phase transitions in physical systems (cf. [Section 3, 7], [11]) and further interpretations of Figure A.4 in such regard are left to the discretion of the reader.

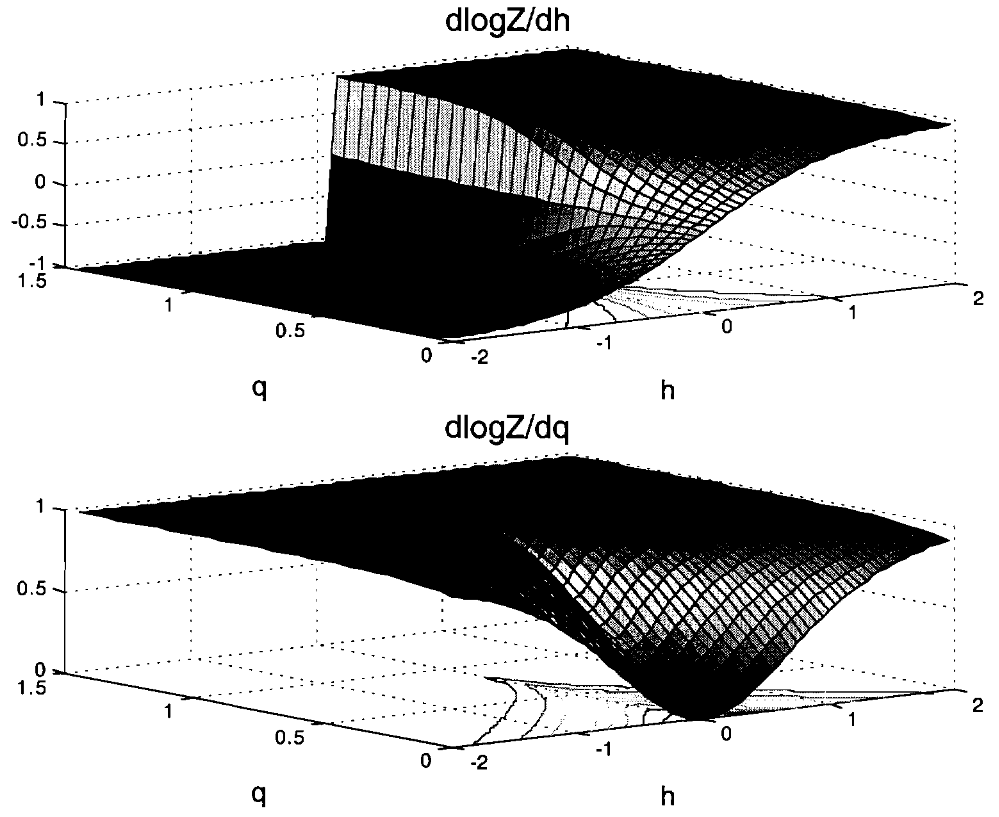


Fig. A.4: Variation of lattice magnetization (top) and correlation (bottom) with respect to q and h.

In Section 3.2, it was asserted that lattice correlation and lattice magnetization can be estimated as sample averages. It is well-known that the sample-average estimators are unbiased, the lower bounds of whose variances can be calculated using the Cramer-Rao inequality [pp. 351,951.

As an extension of the Cramer-Rao inequality, it can be stated that [95] if the probability distribution is of the form such that

$$\frac{\partial \log P_X(X)}{\partial C} = g_1(C)\left[\hat{C} - C\right]$$

holds, then the variance of the estimator $\hat{C}$ attains the lower bound. [pp.306-307, 96], given as $1/g_1(C)$. Here, $g_1(C)$ is known as the (Fisher) Information on C.

Thus

$$\begin{aligned}
\frac{\partial \log P_X(X)}{\partial C} &= \frac{\partial q}{\partial C} \bullet \frac{\partial \log P_X(X)}{\partial q} \\
&= \frac{\partial q}{\partial C}\left[-\frac{\partial \log Z}{\partial q} - \frac{\partial \mathrm{E}(X)}{\partial q}\right] \\
&= N \bullet \frac{\partial q}{\partial C}\left[-\frac{1}{N}\frac{\partial \log Z}{\partial q} - \frac{1}{N}\frac{\partial \mathrm{E}(X)}{\partial q}\right] \\
&= N \bullet \frac{\partial q}{\partial C}\left[-\frac{1}{N}\frac{\partial \log Z}{\partial q} + \frac{1}{N}\sum_{i=1}^{N}\left(\mu_i\mu_{i+1} + \mu_i'\,\mu_{i+1}' + \mu_i\mu_i'\right)\right] \\
&= N \bullet \frac{\partial q}{\partial C}\left[-C + \hat{C}\right] \\
&= \frac{1}{\left(\frac{1}{N^2}\frac{\partial^2 \log Z}{\partial q^2}\right)}\left[\hat{C} - C\right].
\end{aligned}$$

It follows that variance of the estimator $\hat{C}$ is

$$\operatorname{var}\left(\hat{C}\right) = \frac{1}{N^2}\frac{\partial^2 \log Z}{\partial q^2}. \tag{A.3}$$

A similar analysis for the variance of M yields

$$\operatorname{var}\left(\hat{M}\right) = \frac{1}{N^2}\frac{\partial^2 \log Z}{\partial h^2}. \tag{A.4}$$

The value of Equations A.3-4 is in the provision of a means to assess the accuracies of the respective estimators. Note that the respective variances are dependent on the size of the chain N. The plots relevant to Equations A.3-4 are presented below in Figure A.5 (N=1). As stated earlier, the variance of the estimator may equivalently be referred as the inverse of the Fisher information on the parameter being estimated.

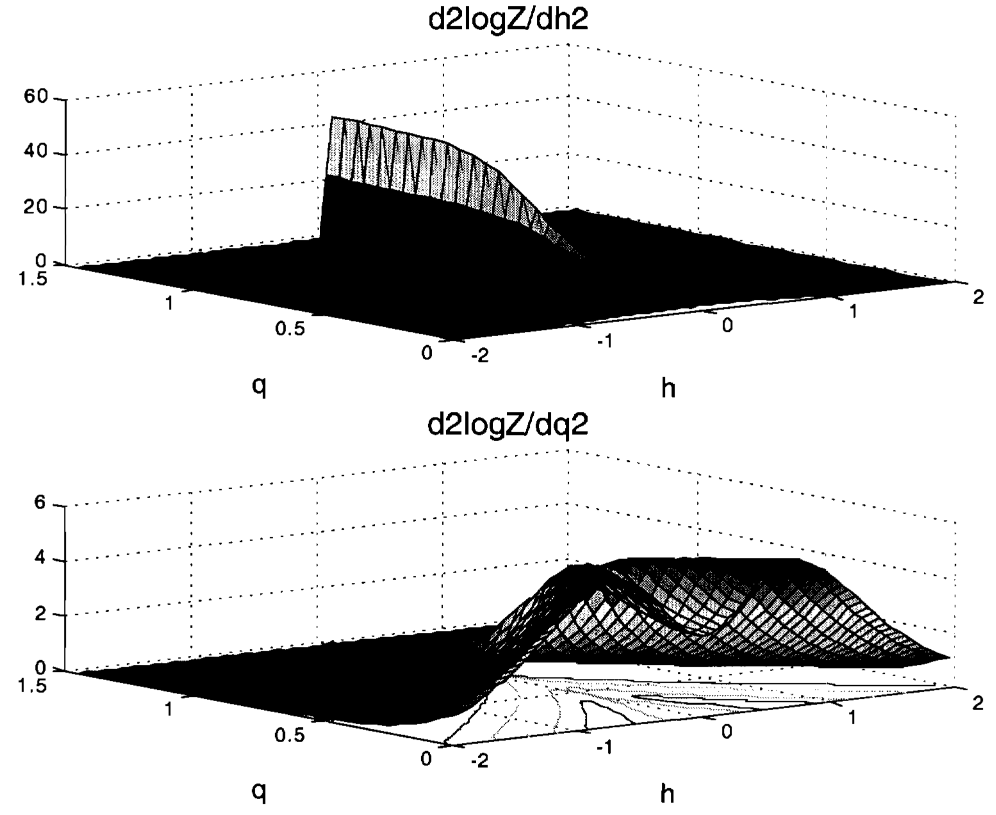


Fig. A.5: Variation of (Fisher Information)-' in the estimation of lattice magnetization (top) and correlation (bottom) with respect to q and h.

As evident from the top plot in Figure A.5, the variance of ***M*** is especially high for $h \approx 0$, for large q. The difference in the original and the simulated images for 'NaturalTurbulence' (cf. Section 4.2.5, Figure 4.9) could possibly be a result of erroneous estimation of the image magnetization.

The MATLAB code for the generation of Figures A.4-5 is listed below.

```
q=0.0001:0.05:1.5;
h=0.0001:0.05:2.0;
h_two=h-1.99;
H=[h_two h];
h=H;
M= length(q);
N= length(h);
for i=1:M,
      for j=1:N,
            temp=model2(q(i), h(j));            %See Appendix A.1
            Lh(i,j)=m_dh(q(i), h(j))/temp;      %See Appendix A.4
            Lq(i,j)=m_dq(q(i), h(j))/temp;      %See Appendix A.5
      end
end
Lh=real(Lh);
```

```
Lq=real(Lq);
%plot magnetization and correlation
subplot(2,1,1);
surfc(h, q, Lh/2);
title('dlogZ/dh', 'fontsize', 16);
xlabel('h', 'fontsize', 14);
ylabel('q', 'fontsize', 14);
subplot(2,1,2);
surfc(h, q, Lq/3);
title('dlogZ/dq', 'fontsize', 16);
xlabel('h', 'fontsize', 14);
ylabel('q', 'fontsize', 14);

figure(2);
[Lhh Lhq]=gradient(Lh, H, q);
[Lqh Lqq]=gradient(Lq, H, q);
% plot variance of estimates (Fisher Information)
subplot(2,1,2);
surfc(H, q, Lqq);
xlabel('h', 'fontsize', 14);
ylabel('q', 'fontsize', 14);
title('d2logZ/dq2', 'fontsize', 16);
subplot(2,1,1);
surfc(H, q, Lhh);
xlabel('h', 'fontsize', 14);
title('d2logZ/dh2', 'fontsize', 16);
ylabel('q', 'fontsize', 14);
```

The remaining Appendices contain detailed listing of computer code. They are available from the thesis of the same name and data or from the authors.

GLOSSARY

Page

$1\times N$ .... ***See*** lattice model
$2\times N$ .... ***See*** lattice model
$3\times N$ .... ***See*** lattice model
a 52
a[illegible] 45
A .... ***See*** transfer matrix
C .... ***See*** correlation
  $C_{1,N}$ 19
  $C_{2,N}$ 21, 27
  $C_{3,N}$ 24
  $\hat{C}$ 28
$\chi_p^{\ 2}$ 62
D_1 60
D_2 60
ΔE 51
E(•) 13
f_r1 65
f_r2 65
F(•) 45
η 60
h 13
$\mathbf{I}_{p\times p}$ 60
l_i 17
L 13
Λ 13, 55
μ 17
  μ_t 18
M .... ***See*** magnetization
  $M_{2,N}$ 2 7
  $\hat{M}$ 28
N 13
p 55.60
$P_{\mathrm{err}}^{\ \mathrm{I}}$ 62
$P_{\mathrm{err}}^{\ \mathrm{II}}$ 62
$P_{\mathrm{err}}^{\ \mathrm{avg}}$ 62
$P_{\mathbf{X}}$(X) .... ***See*** Gibbs distribution
q 13
r_1 65
r_2 65
r(•) 62
T 52
Tr(•) 14
U_i 52
V_i 52
X 13
X 55
  $\boldsymbol{x}_j$ 55
  $\bar{X}$ 59
  $\boldsymbol{X}_{\mathrm{new}}$ 56
  $\boldsymbol{X}_{\mathrm{old}}$ 56
$\boldsymbol{Y}$ 55
  y_j 55
  $\bar{Y}$ 59
Z .... ***See*** partition function
  $Z_{1,N}$ 19
  $Z_{2,N}$ 21,27
  $Z_{3,N}$ 24

A

Axioms of data fusion 88-90

B

Bayes decision rule 56.62.67
Bayes error 62.65
binary image 2.11.43.53.55.57
black-white ratio ....
  .... ***See*** macrocopic properties

C

chain link 49.58
city-block distance 117
Class I data fusion 91.95.99
Class II data fusion 92.95.99.117
class distribution ....
  .... ***See*** macroscopic properties
correlation 7.44
  estimation ....
  .... 28.72.150.152. 153.155.157
  experimental measurements 53.54.71
  formula 14
  solution 19.21.24.27
  tables 29.30-34
  visual interpretation 15
Cramer-Rao inequality 155

D

D.C. flightline data 59.75.97.100
data fusion 85-95
  definition 85.90

decision
- bias 8. 41. 65. 67. 68. 71-73
- functions 68. 69. 71
- graph 99. 137
- threshold 65. 68. 69
- tree 11, 86. 91. 93. 99. 100

DEM .... *See* Digital Elevation Map
detailed balance.................................45
Digital Elevation Map 97, 99. 113

eigenvalue..... 14. 17. 18.21. 24.26. 147
energy (lattice) 9, 13. 14. 57
- computation 51
- function.........................17. 20. 22. 25

energy (spectral)...................55, 97. 102
Expectation Maximization algorithm..58

feature selection...............................103
Forest HYDICE data..........................80
Fisher information.................... 155. 157
fusion suite..98

Gaussian distribution................ 7. 56. 90
Gibbs distribution.................... 9. 13. 56
gradient operator..............................115

homogeneity *See* macroscopic properties
hybrid data fusion................... 92. 93. 95

Ising model 8
- one dimensional 10
- two dimensional 10

lattice model.............................. 2. 7. 13
- $1\times N$....................................15. 18-19
- $2\times N$ (for h=0) 16. 20.21. 29
- $2\times N$ (general) 16. 25.27. 30-39
- $3\times N$.................................... 16. 22-24

lattice configuration 9. 13
lattice energy........... *See* energy (lattice)
looped chain........................... 15. 46. 49

M

macroscopic properties............ 7. 11. 15
magnetization.................................... 14
- estimation........ 28. 149. 154. 155. 157
- experimental measurements...... 53. 54
- formula... 14
- solution 2 7
- tables....................................... 36-39
- visual interpretation........................15

Markov chain....................10. 43. 45. 51
Markov random field........................8. 9
masking.................... 100. 106. 108. 117
maximum likelihood classification75, 80. 98
Metropolis-Hastings algorithm45.46. 52
Monte Carlo method 52. 57
multispectral data ... *See* energy (spectral)

N

negative training........100. 121. 124. 130

P

particle system 1. 8. 13
partition function..................... 9. 13. 14
- computation 19. 21. 24. 27. 154

process model 1. 85. 94. 97

R

random number generator........... 52. 223
regularization theory 7
remote sensing data analysis..1, 7.41. 55
- experiments 75
- process model................................ 94
- decision graph 137

S

scrubbing 95. 100
spectral energy ....*See* energy (spectral)
statistical mechanics.........................7. 9

T

Transfer matrix......................14. 17. 147
- computation 18. 20. 22. 25

Z

zoning... 117