The Application of Maximum Entropy Density Estimation to the Classification of Short Vegetation Using Multifrequency, Polarimetric SAR

by

Yanni A. Kouskoulas

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering)
in The University of Michigan
2001

Doctoral Committee:
Professor Fawwaz T. Ulaby, Chair
Associate Professor Kamal Sarabandi
Professor Demosthenes Teneketzis
Research Scientist Leland Pierce

I dedicate this dissertation to my parents, whom I love dearly.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Fawwaz Ulaby, for his mentoring and support, and for allowing me free rein to take my research in whatever direction it led me. Without him, I would not have come to Michigan for graduate school.

I would also like to thank Dr. Leland Pierce for his lessons on life and research. He has become a mentor and friend; I will not forget his encouragement and support during my graduate career.

No Microsoft products were used in the production of this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

Overview

In this dissertation, we attempt to distinguish between different types of vegetation based on subtle structural differences, using remotely sensed data. This is our overarching goal. There are three major sections:

- The first chapter (which you are reading) is an overview, which describes how the different pieces of the dissertation fit together.

- In Chapter 2, we describes the problem of interest, (called the classification problem) which motivates us throughout. We discusses the starting point for our problem, presenting a useful abstraction and then discuss our sensor, its data, and how we chose to process it. These details are quite crucial to our success.

- Chapter 3 presents a short review of relevant literature and classification tools that are in use today.

- Chapter 4 begins the development of the classifier by presenting a roadmap of classification methods that were developed in our research. We begin with a classifier in the literature, and improve on it. Each subsequent one builds on the previous one, eliminiating weaknesses and building on the strengths of the others, in a quasi-evolutionary fashion.

  By the end of this chapter, our final idea requires us to accurately perform density estimation on our data.

- Chapter 5 discusses the motivation behind density estimation, the literature related to density estimation, and presents the details for the algorithm which

we developed.

We also present extensive tests of the new algorithms, numerically comparing it to the kernel density estimator for a number of oddly shaped densities. We compare accuracy, computational efficiency, and space efficiency.

- In Chapter 6, we are back to thinking about the classification problem. There are some hurdles preventing us from using the density estimation technique in a practical classification problem, and we present ideas and modifications to overcome them.

  We take the classifier in its final form, incorporate the density estimation, and apply it to our data set and the problem at hand. The classification technique we develop is new, and we call it the Bayesian–Hierarchical classifier. We present detailed results of the accuracy of our novel method, and compare it to other methods.

- Chapter 7 is the second to last chapter, and here we present other applications in remote sensing that require the statistical framework and tools we developed.

  We demonstrate methodologies that will lead to accurate modeling of the non–stationary statistics, and apply this to estimating the biophysical properties of short vegetation.

  We also discuss point target representation, and show how current models can be improved by adding background texture to them. To successfully implement this, we develop and demonstrate a deconvolution technique, showing how it

can be used to meet our needs.

- Finally, Chapter 8 is a short summary of our results. Here we describe future work which could extend our ideas.

CHAPTER 2

Introduction to Classification in Remote Sensing with SAR

2.1 Introduction

2.1.1 The Classification Problem

Imagine a satellite, circling the earth, hundreds of kilometers above our heads. This satellite has a synthetic aperture radar (SAR) sensor on it, which takes measurements by repeatedly bouncing electromagnetic signals off of the ground below. It records and processes these signals, slowly building up an image.

Because our sensor is not optical, this image is not a color picture. Rather, each pixel of this image is a vector of numbers, representing the magnitude and phase measurements of the reflected electromagnetic waves for each combination of *receive* and *transmit* polarizations.

The electromagnetic interaction on the ground, and thus the numbers that make up the pixels in our image, are sensitive to the geometry and dielectric constant of whatever is down there. It is because of this sensitivity that our signal contains information about what is on the ground.

Our goal is to analyze this two dimensional image of the earth and be able to identify what we are observing. The difficulty of using this remotely sensed data to identify whether we are looking at the Amazon rain forest or a farmer's wheat fields in Iowa is an example of what we are calling the classification problem.

2.1.2 Utility of Solving the Classification Problem

In the last twenty years, the scientific community has put a great deal of effort into solving the classification problem, developing myriad techniques that allow automatic

classification of remotely sensed data.

With all sensors that look down at the earth, from SAR to optical imagery, the most basic question we can ask is, "What is down there? Are we looking at a city, suburbs, or countryside? If there is vegetation, is it agricultural, or a forest of trees?" We can ask more specific questions, also, such as "What kind of plants are growing down there?"

If we cannot accurately answer these questions, we will be unable to answer the next set of more difficult questions, such as, "How tall is this building?" or "How much plant biomass is down there?" or "What is the moisture of the soil underneath the plants?"[1]

Answering these questions can be useful for various applications, such as

- Mapping the distribution of vegetation on the earth on a global scale. (This application is called land cover classification.)
- Monitoring wetlands or forests to understand how they change over time.
- Monitoring natural disasters like avalanches or forest fires and how they change the earth.

All these kinds of monitoring help us better understand the global ecosystem.

In the case of avalanches and forest fires, SAR information could be useful for real time crisis management, because it can see through clouds, smoke, and sometimes trees. It could give a clear picture of a changing situation that might otherwise be obscured.

[1] Or, for the military types, "Is there a tank in that forest?"

In addition to these applications, classification is the first step in estimating underlying biophysical parameters of vegetation, such as soil moisture, plant biomass and plant height.

Estimating biophysical parameters opens up another set of applications. For example, estimating soil moisture is of interest to hydrologists, in helping predict the weather and understand the water cycle. Estimating crop biomass is of interest to farmers, who might eventually be able to use it to help them predict crop yields. In turn, predictions of crop yields could lead to the ability to predict shortages or surpluses of food on a countrywide scale, and thus could lead to the prediction and possible prevention of famine.

Although all of the aforementioned examples are product-oriented end applications, this kind of measurement also has scientific value. From this perspective, classification can be a first step in electromagnetic modeling of vegetation. This modeling is no mean feat, since vegetation is a very complicated, non-deterministic target whose statistical properties have never been adequately characterized.

2.1.3 Why Use SAR?

SAR is an excellent candidate for doing classification because:

- It is affected by geometry, and we want to classify the ground based on geometrical differences.
- It can take tremendous amounts of data very quickly, and give us rapid coverage of the entire earth.

- It can take data continuously, at night, and through clouds.
- SAR is theoretically range and sensor independent, so a measurement taken 20 meters off the ground with sensor A gives the same results as a measurement taken 200 kilometers off the ground with sensor B.

2.1.4 Why Is Classification So Difficult?

The challenge in the classification problem is to be able to distinguish between different areas on the ground, based on what our sensor is measuring.

It should be noted that the ability to use SAR data to distinguish between water and trees is something that has been reliably demonstrated in the literature. In cases like these, the classification problem has been solved by a variety of the methods discussed in the literature review, in Chapter 3. However, classification becomes much more challenging when we attempt to classify targets based on very fine geometrical differences, such as distinguishing between different types of ice or distinguishing between different types of short vegetation.

Because of these differences, the classification literature distinguishes between the different difficulties of classification problems. Level one classification typically separates short vegetation from tall vegetation from water from bare soil. Level two classification further subdivides the level one classes, so that instead of tall vegetation, level two would identify different types of trees. Instead of short vegetation, level two would identify the different types of short plant types on the ground.

In this research, we concentrate on level two classification of short vegetation.

Rather than just identify that an area of ground is covered by short vegetation, we want to be able to identify the structural class that we are looking at.

Just what makes level two classification of short vegetation so difficult?

Figure 2.1: A SAR image composite of L and C band radar data, where L band signal strength is represented by the color purple, and C band signal strength is represented by green.

Taking a rather simplistic view, there are no standard visual cues in a SAR image to tell us what we are looking at. We cannot see the shape of the leaves, or the height of the vegetation. The untrained observer would find it nearly impossible to interpret a radar image, and even the trained observer would have trouble identifying types of

vegetation. The reader might consider Figure 2.1, and try to identify what we are looking at. Someone who is familiar with SAR images could identify water (dark area) trees (bright area) and short vegetation (medium brightness areas). However, the same expert would have trouble identifying specific types of trees or the type of short vegetation.

Consider Figure 2.2, where we take a more quantitative approach, plotting values of the strength of the radar backsacttering coefficient σ^o_{CHV},[2] on four different days.

We still have difficulties. Not only does the measurement of wheat in one field have the same value as the measurement of corn in another field for day two, but the relationships change over time, making a technique developed on a radar image that works for day one, perform poorly on the second measurement day.

This variation in a particular field from day to day occurs, to a lesser extent between different fields on the same date.

The error bars on this plot indicate the variance in the measurement within a particular field. Complicating matters further, there is a natural sensor noise called fading, (which we will discuss in Chapter 3) which is not shown in this plot, and increases the size of the error bars significantly.

The variations in our measurements come from sources such as the variation of biophysical parameters, the change in structure of a type of vegetation over the growing season, and the difference in soil moisture between different areas of the ground.

[2]The different aspects of radar data acquisition and processing will be discussed in detail in Sections 2.2.4 and 2.2.5.

The notation here indicates that the measurement is a radar backscattering coefficient whose frequency was C-band, and whose receive and transmit polarizations (respectively) are horizontal and vertical, with respect to the coordinate system of the SAR.

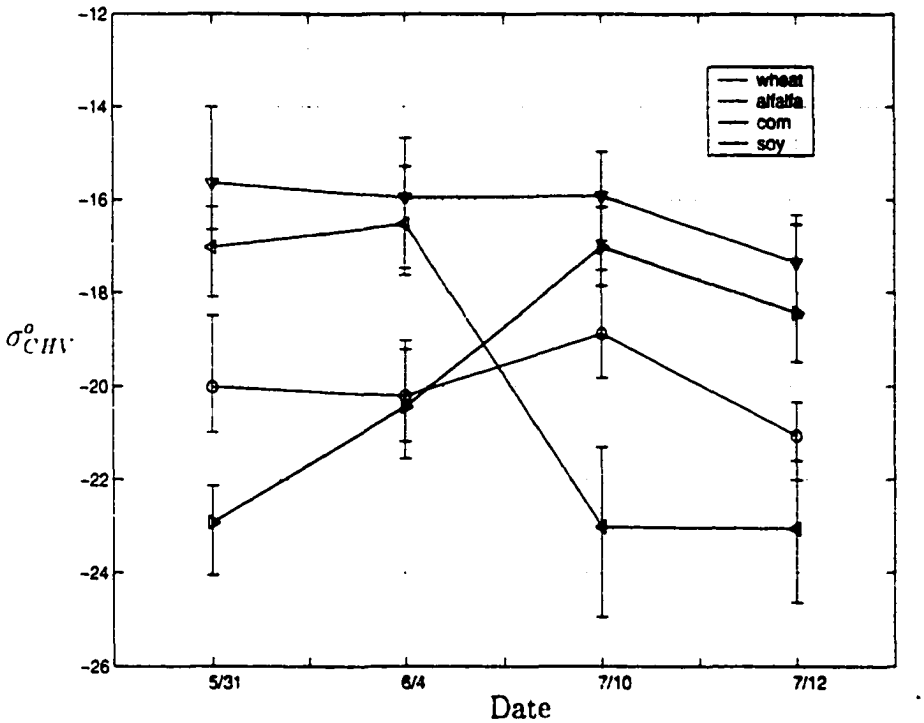


Figure 2.2: Wheat, alfalfa, corn and soybean radar measurements over four different days. Error bars indicate the dynamic range of measurements within a single field. This plot does not include the noise due to fading, which would increase the error bars significantly.

These uncertainties in our measurements will be carefully discussed in Section 2.2.1.

The conclusion we can draw from this is that SAR measurements of vegetation vary as much or more within a particular field or over a series of days than they do between different plant type. This variation is one of the challenges that we have before us, and one that makes the classification problem difficult.

Although we are concentrating on classifying short vegetation, the proposed classification techniques have wide applicability and can, with no modifications, be applied to other areas of remote sensing.

2.2 Data

Consider our original example, of a sensor flying high above the ground, bouncing an electromagnetic wave off the vegetation canopy below, as in Figure 2.3.

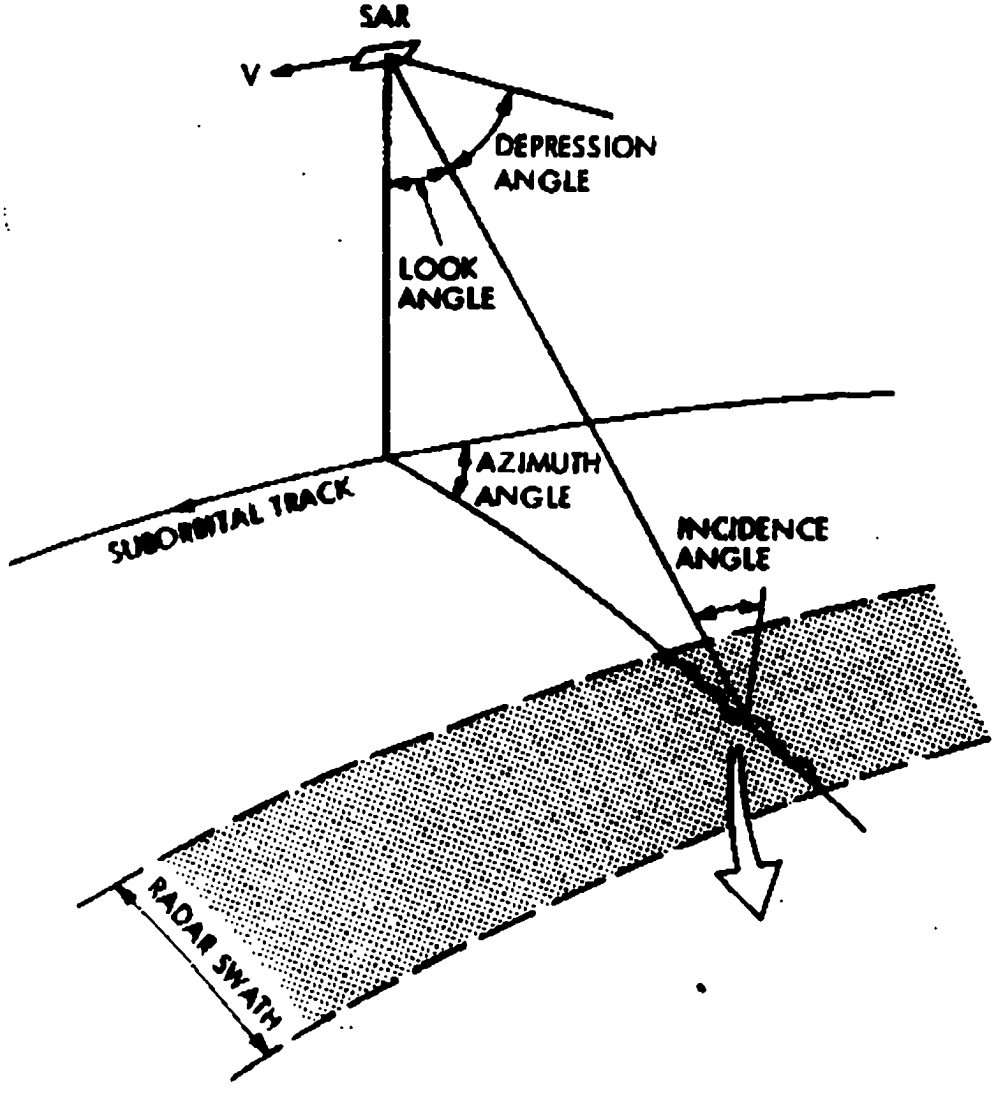


Figure 2.3: A SAR measurement system.

The total electromagnetic wave that is reflected back from our vegetation canopy consists of the summation of the reflections from many randomly distributed scatterers, whose properties (strength, phase, spatial distribution, polarimetric response) all

depend on the dielectric properties and the geometry of the vegetation. The dielectric properties and geometry of the vegetation depend on (among other things) the biophysical parameters of the vegetation, such as height, stage of growth, biomass, and sometimes soil moisture.

In Figure 2.2, we saw the difficulties of classifying short vegetation with SAR. The crux of the problem is that the electromagnetic measurements of the same crop in different fields or on different days can differ more than two different crops side by side. Our measurements have large, overlapping dynamic ranges.

The subsequent sections discuss the reason for our measurement uncertainty, and the acquisition and processing of a set of radar images for the purposes of developing and testing a classification algorithm.

2.2.1 Measurement Uncertainty

To be successful at classification of short vegetation, we must understand the constant fluctuations in the measurements, over time and for different areas of the ground. There are four major reasons for this variation between measurements.

1. The first reason for the signal fluctuation is fading. Fading is sometimes called speckle or scintillation. These fading effects are a consequence of our coherent sensor, and behave like an unavoidable layer of noise superimposed on our data.

   Fading effects occur when our transmitted electromagnetic wave reflects off the many randomly distributed scatterers which make up the vegetated terrain. The multiple reflections constructively and destructively interfere with each other,

and give rise to a statistical uncertainty in the signal level. This uncertainty or fluctuation is so large that two independent measurements of the same class with the same input conditions $\boldsymbol{S}$ can have signals that differ by orders of magnitude.

2. The second reason for our signal fluctuations are the quantified changes in the temporal and spatial domain of the vegetation geometry and dielectric constant.

   This signal fluctuation corresponds to a change in the biophysical parameters or moisture of the vegetation over time and space.

   Our measurements of biophysical and other parameters affecting the electromagnetic radar measurement (such as biomass, height, crop type) are collectively called ground-truth data. Collecting ground-truth information can be as simple as going to the field and identifying the vegetation type or as complicated as measuring height, biomass, plant spacing, counting the average number of leaves on the average plant, and taking cores of the soil to measure soil moisture in different parts of the field.

   Thus, if we are measuring the average wet biomass each time we take an electromagnetic measurement, and this parameter changes from one field to another, causing the requisite change in the reflected electromagnetic signal, we could say that this is a result of changes in the ground-truth measurements.

3. The third reason for our signal fluctuation are the unquantified changes in the temporal and spatial domain of parameters of the vegetation geometry and dielectric constant.

This is exactly the same as the second item, except it is at a finer scale, or it is a parameter that we have not quantified with our ground truth measuremnts.

No two plants are ever quite the same, and in one area of the field, the plants might be leaning to one side, leaving a hole in the foliage. Right next to such hole, a group of plants could be clustered together, creating an area that has many leaves clustered together.

4. The fourth reason for the fluctuation in our signal is system error. This accounts for the invariable differences in our measurement systems because of their design, or differences in the same measurement system because of temperature, operation, or other factors.

   Fortunately for us, the process of calibrating a SAR removes effects associated with the individual system. As long as our data is calibrated correctly, we can safely neglect this fourth source of variability in our measurements.

The distinction between the second and third items on this list is a fine one, and depends to a great extent on what input parameters we measure, and how carefully we measure them.

In theory, if we could take enough ground-truth data accirately enough and at a fine enough spatial scale, we might be able to eliminate the effects of unquantified changes in the vegetation on our signal. Our ground-truth measurements would consist of continuously varying maps of soil moisture and spatial distribution of leaf density and stem angle. In reality, even if we take measurements as carefully as possible, no one has ever completely characterized the statistical properties of the plants

themselves. For this reason, we retain the distinction between quantified variations in the vegetation parameters, and unquantified variations in the vegetation parameters.

2.2.2 Abstraction

Abstracting the classification problem–stating it in a generalized way, or without reference to the specific case we are dealing with–gives us two very useful advantages.

Firstly, there are many problems which, when generalized, become equivalent. Solving the abstracted version of the problem automatically solves the whole set of specific problems.

Secondly, the generalization allows us to see the problem with only the essential features. This makes it easier to model and quantify the situation, concentrating on the core of the problem rather than on the details. This abstraction lets us see our problem in a new light, and directs us towards a different approach to solving it.

Our sensor, as discussed in Section 2.1.1 produces a two dimensional radar image. In Figure 2.4 we represent each pixel of the image as a system. Here, a system is a process that transforms input signals to output signals. The system describing each pixel may be complicated, nondeterministic, continuous time, non-linear and non-stationary. When we discuss "the measurement system" or the "system that produces our data set," we are referring to Figure 2.4. When, in the course of our discussion of SAR, we write "inputs" or "outputs" we are also referring to Figure 2.4, and identifying the inputs or outputs of the radar measurement process.

The system inputs, all taken together, form a vector of numbers that describe

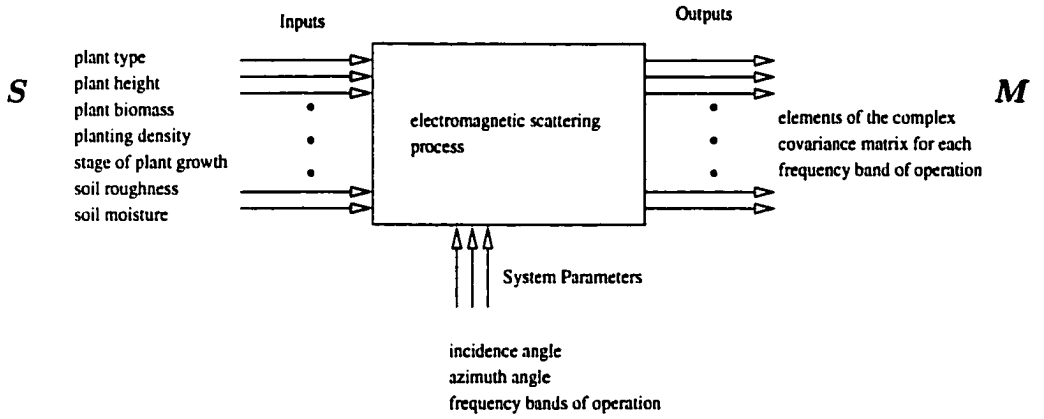


Figure 2.4: A SAR pixel from a systems perspective

the quantified ground-truth information (usually consisting of biophysical parameters averaged over an area) about the vegetation at the time the pixel was imaged. We will call this vector of inputs, $\boldsymbol{S}$. For the remainder of this dissertation, the terms input parameters, inputs and ground-truth data will be used interchangeably.

Similarly, the system outputs can be grouped together into a vector that describes our measurement. We will call the output vector $\boldsymbol{M}$. The elements of the input vector $\boldsymbol{S}$ are the identity of the pixel and its physical characteristics, while the elements of the output vector $\boldsymbol{M}$ are what the sensor actually measures. Elements of the output vector are sometimes called channels or dimensions of the measurement, and these terms will be used interchangably.

We can now imagine that each vector $\boldsymbol{S}$ represents a point in a multidimensional input space. Similarly, each vector $\boldsymbol{M}$ is a point in a corresponding output space. The 'black box' system described above creates a mapping between the input and output vectors (and thus their corresponding multidimensional spaces).

For each measurement we can take the output vector and plot it in the output

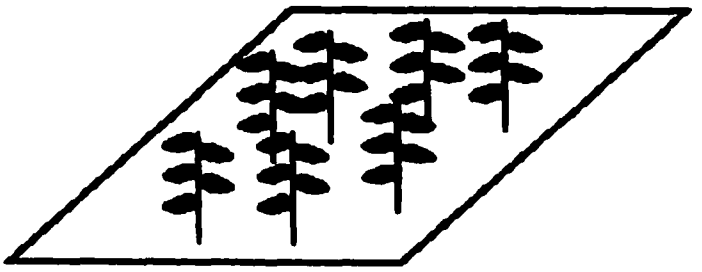


Figure 2.5: Statistics of multiple plots with fixed inputs in a SAR image.

space, with each element of the vector being plotted along an axis. Thus a set of measured data would be represented by a cloud of points in the multidimensional output space.

2.2.3 Definitions

Using what we now know about the uncertainty in our data, we will propose some definitions which allow us to refer more easily to the effects we see in our data sets.

Statistics of a Single Pixel

Consider what happens if we point our sensor at a single plot of vegetation of class 'c' (Figure 2.5) in which all the biophysical parameters that affect the radar measurement are fixed. We take a measurement, and then fly our sensor to another plot of ground, where the biophysical parameters are in some way fixed to be exactly the same as before. If we repeatedly take measurements of different plots of vegetation with fixed biophysical parameters, we will build up a data set where the uncertainty is due only to speckle.

Assume that $\boldsymbol{\mu}$ and $\boldsymbol{m}$ are vectors of numbers that represent the measurements of backscattered electromagnetic waves. The measurement $\widehat{\boldsymbol{M}}$ is the constant value

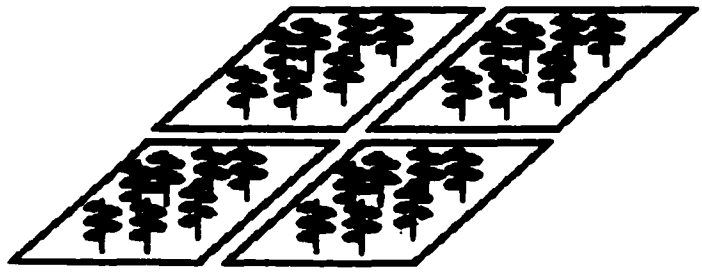


Figure 2.6: Statistics of many pixels in a SAR image.

that we would measure if there were no speckle. We define $\boldsymbol{M}$ to be a measurement of our data in the presence of speckle.

We represent the variation due to speckle by defining a conditional probability density function $K(\boldsymbol{M} = \boldsymbol{m}|\widehat{\boldsymbol{M}} = \boldsymbol{\mu})$. This represents the probability that we measure $\boldsymbol{m}$ when the underlying measurement is $\boldsymbol{\mu}$. Note that the fading statistics are described by a known, class independent process [39].

Statistics of Many Pixels

Again, we take measurements of the ground which is covered with class 'c'. While collecting this next data set, we let our sensor move around in a more realistic fashion so it is measuring different areas of the ground, and we let time pass so that the input parameters change (Figure 2.6).

The uncertainty in this data set comes from both speckle and the temporal and spatial variations of the biophysical parameters of the vegetation, at all scales. Define $r_c(\boldsymbol{M} = \boldsymbol{m})$ as the density function which describes the statistics for the raw data set that we just measured.

We can also define $t_c(\boldsymbol{M} = \boldsymbol{m})$ as the density function which describes the statistics for the raw data due to spatial and temporal variation of the biophysical param-

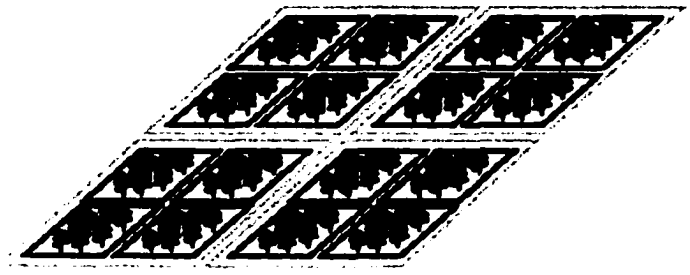


Figure 2.7: Statistics of filtered (averaged) pixels in a SAR image.

eters of the vegetation, if there were no speckle in our image. Note that $t_c(\boldsymbol{M} = \boldsymbol{m})$ is not directly measurable, because of the nature of our sensor.

Statistics of Averaged Pixels

Finally, we create a data set which takes measurements of class c from many different dates, times, and places. We filter the data, combining adjacent pixels to reduce the fading in our data (Figure 2.7.) Every set of adjacent pixels that are combined together is called a stand.

How many pixels should we combine to create a stand? The more spatial averaging we do, the less uncertainty due to fading. However, if we do too much spatial averaging, we lose relevant texture information. We want to reduce fading to a fraction of the total dynamic range of our vegetation measurements, while also preserving texture at the scale that is useful for classification. We will see that the variation due to texture can be useful, while the variation due to fading is not.

For this data set, the uncertainty is due mainly to inter-stand texture. The filtering process minimizes (but does not totally remove) the uncertainty due to speckle and inter-pixel texture.

We define $f_c(\boldsymbol{M} = \boldsymbol{m})$ as the density function which describes the statistics of a

single stand, which consists of the filtered data for class 'c'.

2.2.4 Acquisition

In this section we will discuss the actual electromagnetic measurements, which we have been representing with a vector $\boldsymbol{M}$. We will review what they mean, where they come from, and how we chose the data for this investigation.

Of all the data available to us, we have chosen to use data from Jet Propulsion Labs' AirSAR sensor, which is a synthetic aperture radar mounted on an airplane. The set of data that we used was the best we could find that had comprehensive measurements of ground truth, many different types of vegetation in different areas and coverage of different days.

We used AirSAR measurements of the Kellog Biological Station (pictured in Figure 2.8) in Kalamazoo, Michigan taken in 1995. This area had plots of 'homogeneous' agricultural vegetation, and was accessible to us so that we could make ground-truth observations. The data was taken on different dates throughout the growing season to ensure that we measured a wide range of possible conditions that the plant could be in. During the time that the SAR was taking measurements overhead, we measured ground-truth parameters such as soil moisture and the height of vegetation on the ground.

We selected calibrated SAR swaths with fixed azimuth and incidence angles.

The simplest way to model what we are measuring with a synthetic aperture radar (or any backscattering radar-based sensor, for that matter) is to think about

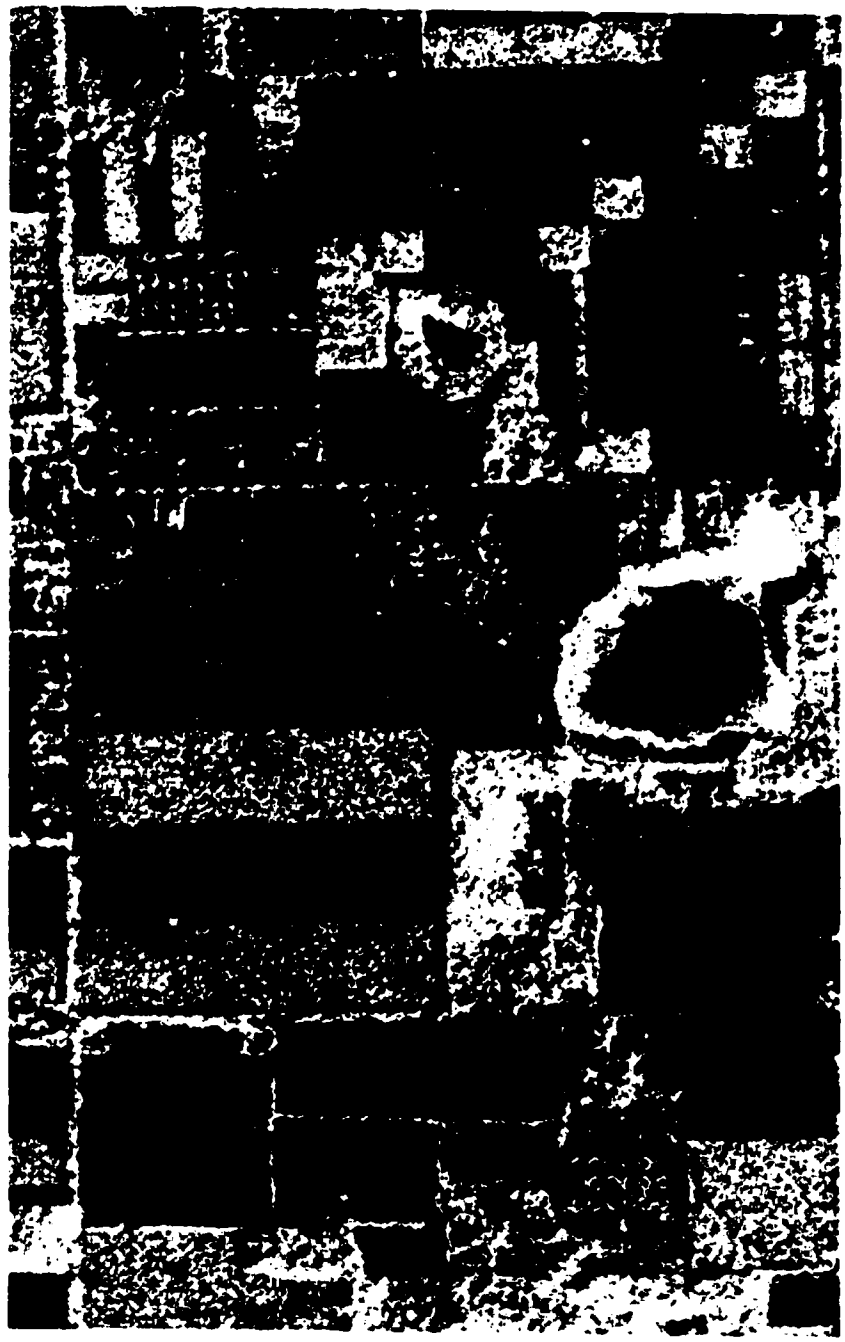


Figure 2.8: A birds eye view of Kellogg Biological Station

the incident electromagnetic field being transmitted from the sensor, bouncing off the target, and reflecting back, to be measured by the sensor. If the incident field is $\boldsymbol{E}^i = \boldsymbol{v}E_V^i + \boldsymbol{h}E_H^i$ and the scattered field is $\boldsymbol{E}^s = \boldsymbol{v}E_V^s + \boldsymbol{h}E_H^s$, we can relate the two fields by writing,

$$\begin{pmatrix} E_V^s \\ E_H^s \end{pmatrix} = \frac{e^{-jk_o r}}{r} \begin{pmatrix} S_{VV} & S_{VH} \\ S_{HV} & S_{HH} \end{pmatrix} \begin{pmatrix} E_V^i \\ E_H^i \end{pmatrix} \tag{2.1}$$

Where H and V subscripts indicate horizontal or vertical polarization. The variable r is the distance from the sensor to the target, and k_o is the wavenumber of the incident electromagnetic wave. All these variables follow the conventions in [38]. The matrix which relates the incident and scattered fields is known as the scattering matrix.

We can use a radar system to measure the incident and scattered fields, and with the knowledge of r from the signal delay, the elements of the scattering matrix can be obtained readily. These elements, rather than being sensor dependent, represent characteristics of the terrain.

Consider a measurement of a particular scattering matrix element for a single pixel of our image. We will find the measurement is a complex number, which we can plot as a single point on the complex plane.

If we assume that an area of ground is a homogeneous distributed target, such as vegetation, we know that repeated measurements of this area of ground will give different results, because of the fading processes involved. Although the scatterers in our measurements are statistically similar, for each measurement they are randomly

placed. Their contributions to the backscatter add together, and due to the Central Limit Theorem, the final result is a jointly Gaussian distribution. Both the real and the imaginary part of the scattering matrix element are independent and identically distributed.

Thus, taking multiple measurements of this pixel would give us a cloud of points centered at the origin: a zero–mean Gaussian, with a particular covariance matrix, given as follows.

$$\begin{pmatrix} \langle S_{HH}S_{HH}^* \rangle & \langle S_{HH}S_{VV}^* \rangle & \langle S_{HH}S_{HV}^* \rangle & \langle S_{HH}S_{VH}^* \rangle \\ \langle S_{VV}S_{HH}^* \rangle & \langle S_{VV}S_{VV}^* \rangle & \langle S_{VV}S_{HV}^* \rangle & \langle S_{VV}S_{VH}^* \rangle \\ \langle S_{HV}S_{HH}^* \rangle & \langle S_{HV}S_{VV}^* \rangle & \langle S_{HV}S_{HV}^* \rangle & \langle S_{HV}S_{VH}^* \rangle \\ \langle S_{VH}S_{HH}^* \rangle & \langle S_{VH}S_{VV}^* \rangle & \langle S_{VH}S_{HV}^* \rangle & \langle S_{VH}S_{VH}^* \rangle \end{pmatrix} \tag{2.2}$$

We wish to characterize the complex covariance matrix of this pixel, because this completely characterizes the statistics of the backscattered electromagnetic wave. The better we estimate each element of the covariance matrix, the more we reduce fading in our measured data.

If we are given a single-look measurement of the scattering matrix elements of a particular pixel, we must realize that any estimate of the pixel's covariance matrix will necessarily be poor.

What we will do is take a group of adjacent pixels, and ensure that they come from the same vegetation type and that their input parameters are the same. Thus, the only variation in this measurement is from fading, and we can treat these pixels as independent samples of our target, for the purposes of estimating the covariance matrix. We are assuming that the textural variation of the vegetation is spatially

distributed so that it is at a large scale with respect to the samples we are taking. If this is the case, the texture is constant for the small area of pixels we are averaging.

After calibration, the data we have chosen does not come in the form of a scattering matrices or covariance matrices. Data from each pixel in each frequency band of the SAR image comes in the form of ten 8-bit numbers which are a compressed representation of what is called the JPL Mueller matrix.

The JPL Mueller matrix consists of linear combinations of the terms in the covariance matrix, such that the covariance matrix elements can be recovered with simple algebra. Thus, these two forms are equivalent representations, in that they both contain the same information: one can be reconstructed from the other.

The Mueller matrix and its derivatives are analogous to the scattering matrix, in that they relate the incident and scattered electric fields when the fields are represented as Stokes vectors. JPL uses what is known as the JPL Stokes vector representation for their electric fields, which can be expressed as [38],

$$F_{JPL} = \begin{pmatrix} \langle |E_H|^2 \rangle + \langle |E_V|^2 \rangle \\ \langle |E_H|^2 \rangle - \langle |E_V|^2 \rangle \\ \langle -2\Re(E_V^* E_H) \rangle \\ \langle -2\Im(E_V^* E_H) \rangle \end{pmatrix} \tag{2.3}$$

This definition leads directly to the JPL Mueller matrix, which relates the incident and reflected Stokes vectors. The JPL Mueller matrix (M_{JPL}) is a 4×4 symmetrical matrix whose elements are defined in terms of scattering matrix elements as follows:

$$M_{11} = \frac{1}{4}(S_{HH}S_{HH}^* + S_{VV}S_{VV}^* + S_{HV}S_{HV}^*) \tag{2.4}$$

$$M_{12} = \frac{1}{4}(S_{HH}S_{HH}^* - S_{VV}S_{VV}^*) \tag{2.5}$$

$$M_{13} = \frac{1}{2}\Re(S_{HH}S_{HV}^* + S_{VV}S_{HV}^*) \tag{2.6}$$

$$M_{14} = \frac{1}{2}\Im(S_{HH}S_{HV}^* + S_{VV}S_{HV}^*) \tag{2.7}$$

$$M_{22} = \frac{1}{4}(S_{HH}S_{HH}^* + S_{VV}S_{VV}^* - S_{HV}S_{HV}^*) \tag{2.8}$$

$$M_{23} = \frac{1}{2}\Re(S_{HH}S_{HV}^* + S_{VV}S_{HV}^*) \tag{2.9}$$

$$M_{24} = \frac{1}{2}\Im(S_{HH}S_{HV}^* + S_{VV}S_{HV}^*) \tag{2.10}$$

$$M_{33} = \frac{1}{2}\Re(S_{HV}S_{HV}^* + S_{VV}S_{HH}^*) \tag{2.11}$$

$$M_{34} = \frac{1}{2}\Im(S_{HV}S_{HV}^* - S_{VV}S_{HH}^*) \tag{2.12}$$

$$M_{44} = \frac{1}{2}\Re(S_{HV}S_{HV}^* - S_{HH}S_{HV}^*) \tag{2.13}$$

This format is documented in [1]. The data that Jet Propulsion Labs provided us with were calibrated images where each pixel was a compressed version of the JPL Mueller matrix format.

2.2.5 Processing

The purpose of processing the data was convert it to a form that was useful and easily applied to the problem at hand. As described above, we wanted data that characterized common plant geometries, and we wanted the data to encompass a wide range of the possible states of moisture, height, and geometry and other inputs.

We are not interested in the whole image at this point, because we do not have

ground-truth measurements or even know what is growing in most of it. We would like to take the areas that we do have ground truth for, and convert the pixels of this image into a set of multidimensional data points. Corresponding to each of these data points, we will have information on plant type, plant height, and some other ground truth that was taken. To convert the data into a convenient form, we designed a data processing path for our images. The input to the data processing path was a calibrated SAR image from JPL. The ultimate goal in terms of output was to create a multidimensional data point for each stand of our distributed target.

There are many transformations that one could apply to the mueller matrix representation, and although we would like to reduce its dimensionality, we want to preserve information we have about the electromagnetic measurement. For example, we need to be cautious about applying a transformations such as polarimetric synthesis, since although it reduces the dimensionality of our measurement, we lose information about the response from different polarizations.

To this end, we represented our data as backscattering coefficients of each available polarization (to represent the magnitude) and phase statistics (to represent the difference in phase between scattering matrix elements) of our signal.

The backscattering coefficient for *receive* polarization p and *transmit* polarization q is defined as follows:

$$\sigma^o_{pq} = \frac{4\pi}{A}\langle|S_{pq}|^2\rangle \tag{2.14}$$

where A is the area on the ground that the pixel represents and S_{pq} is the scattering matrix element.

The co-polarized phase statistics are defined by two parameters,

$$\alpha = \frac{|\langle S_{VV} S_{HH}^* \rangle|}{\sqrt{\langle |S_{VV}|^2 \rangle \langle |S_{HH}|^2 \rangle}} \tag{2.15}$$

$$\zeta = \tan^{-1} \frac{\langle \Im(S_{VV} S_{HH}^*) \rangle}{\langle \Re(S_{VV} S_{HH}^*) \rangle} \tag{2.16}$$

The cross-polarized phase statistics were also computed, but after some initial experimentation, we found them not to be useful for classification purposes.

An excellent explanation of the derivation of these parameters and the assumptions behind them can be found in [29]. The advantages in reduction in dimensionality of a data point and resulting simplification of the data set that can be afforded by this representation are significant, mainly because the new representation preserves useful information needed for classification.

Taking reciprocity into account, a covariance matrix of the scattering matrix elements can be represented as a nine–dimensional data point. If we represent the measurement in terms of backscattering coefficients and co-polarized phase statistics, we are representing it as five dimensional data point.

By converting the JPL Mueller matrix representation to backscattering coefficients and phase statistics, we are incorporating additional information about the sensor into our model, and thus reducing the dimensionality of the data without loss of information.

One of the problems that we have with our data is fading, and at this point in the data processing path, we make an effort to reduce fading to manageable levels. We would like a high enough number of looks to minimize fading (get a good estimate of the covariance matrix parameters), but with enough samples to give us the natural

texture variation of the target.

To fulfill this criterion, we filtered the image (using direct spatial averaging of the Mueller matrices), trading off resolution for a higher number of looks.

The slant range and azimuth resolutions for the AirSAR, as given in the data headers was 5.5m, and 1.2m, respectively. Thus, ground range for the incidence angle we chose to work with (45°) is 7.8m. After warping (a process which will be explained shortly), which performed nearest neighbor resampling, the pixel dimensions in our data were 5.9m and 5.0m, for the x and y dimensions, respectively. We estimated this by counting pixels in one field with known dimensions.

We chose to group our data into stands (as discussed in Section 2.2.3) which were squares with 5 pixels on each side. Averaging each 5×5 square meant that each sample contained an area of $[5(5\text{m})] \times [5(5.88\text{m})] = 735\text{m}^2$. Since one statistically independent sample covered $(7.8\text{m})(1.2\text{m}) = 9.3\text{m}^2$, we estimated the number of looks by dividing the area averaged by the area of one look. Thus, we computed that using this type of filtering, we would get 78 looks per data point.

After reading the file from magnetic tape, the processing proceeded as follows:

1. Modify the image headers and sizes so they became identical. JPL headers contain information about the image contained within, which can, at times, be misleading. For the L and C band files we synchronized the file sizes, number of lines in the image, and the latitude and longitude of the images.

2. Fit each image to a map, removing all perspective effects. This process is called warping, and was alluded to previously. We needed to do this because each

image is initially a rectangular strip of pixels, not necessarily oriented in any direction. After fitting it to a map, data could be extracted from each image the same way, using a cookie cutter type approach.

For this task, we used PCI, which is a collection of image processing software that include common image manipulation tools. We used a part of PCI called GCPWorks to associate points in the SAR image to points on an accurate elevation map of the area (also called a digital elevation map or DEM). Each association of a point in the radar image to a point on the DEM is called a ground control point.

After invoking GCP Works and manually identifying ground control points, we performed further processing to prepare the data for running the map-fitting code called REG. REG warps the SAR image to the map to remove perspective effects, stretching it like a rubber sheet on a table, to fit our map.

The only improvement we could have made in this fitting was taking the height dimension into account. However, because the area of interest is relatively flat, this was not a serious issue.

3. Draw bitmaps indicating each area of interest. Their coordinates were noted, and each image was checked visually to ensure accurate placement of the bitmaps. In addition, the visual check allowed us to remove any stray point targets, edges, and other artifacts which made our data anything other than pure distributed target.

4. Decompress the JPL Mueller matrix for all the pixels in the specified field,

average them (increasing the number of looks as described above), and generate scattering coefficients and phase statistics.

To do this, we developed a piece of code (called FIELD) which ran within the framework of PCI. When fed the coordinates of a rectangular field, FIELD decompressed the JPL Mueller matrix, processed it as specified, and returned the scattering coefficients and phase statistics.

Dates in 1995: May 31, June 2, June 4, July 10, July 12, July 14
Sensor: JPL AirSAR
Site: Kellog Biological Station, Kalamazoo, MI, U.S.A.
Channels: L and C band, polarimetric data
Looks per data point: 70

Crop	# Points	Vegetation Heights	Vegetation Biomass	Soil Moisture
Alfalfa	1845	10-60cm	0-5.6 kg/m^2	0.14-0.33
Corn	864	93-118cm	0-2.0 kg/m^2	0.11-0.17
Wheat	1260	49-103cm	1.6-3.1 kg/m^2	0.14-0.35
Bare	909	0cm		0.15-0.18
Soy	360	$\star$	$\star$	$\star$

Table 2.1: Data summary

Because of the complexity of this task, we automated it. Using the coordinates for each field, we created a script which ran FIELD for all the different crop types, and generated matrices in the form of a matlab readable text file.

After processing, the data gave a good statistical sampling of five very pure structural classes of vegetation: corn, wheat, alfalfa, soybeans, and bare soil. Each data point was labeled with the vegetation type growing there at the time the image was taken. Each data point had ten independent measured parameters which were usable in classification. These parameters make up the measurement vector (which in Fig-

ure 2.4 we called M) and consisted of backscattering coefficients (σ^o_{HH}, σ^o_{VV}, σ^o_{HV}) and the co-polarized phase statistics (α, ζ) for each frequency band. Figure 2.1 is a summary of the processed data. Note that the starred row ($\star$) is soybeans, which we had no ground truth measurements for. However, the data taken for this field was taken throughout the growning season, on the same days that the rest of the data was acquired.

2.3 Assumptions and Class Selection

In order to develop a useful classification algorithm, we must reconcile the capabilities of the sensor with the needs of the user. The electromagnetic processes that a SAR depends on are sensitive to geometry and dielectric constant, so these are the parameters that our data will be sensitive to. On the other hand, the user might like the genus and species of the vegetation that we measure.

We must establish compatibility between the user perspective and the sensor perspective. Our technique should ultimately be sensitive to dielectric constant and geometrical differences between different vegetation types, but must not be confused by changes in dielectric constant or geometry within a given class. Choosing classes incorrectly based on insufficient understanding of what our senor responds to will lead to low accuracies in our classification.

We chose a measurement site which afforded us access to five different types of short vegetation: wheat, soybeans, alfalfa, bare soil, and corn. From now on we will refer to these groupings as our classes. All have different geometries, and the implicit

assumption is that the crops chosen are representative of their respective geometric categories.

This means that when we say we are identifying 'wheat,' we are really identifying a grass-like geometry, so a field of wild grass would likely behave similarly to a field of wheat, from the point of view of the SAR. Similarly, if we identify a region of ground as 'alfalfa,' it means that the vegetation there has the geometry of a herbacious shrub.

Furthermore we assume that our data encompasses a wide range of the possible conditions that exist for this class. For a given class, the biophysical parameters that affect our sensor can vary widely over time and space. The vegetation height changes, the soil moisture changes, and the plant structure changes, among other things. We would like, insofar as possible, for our data to reflect most of the combinations of conditions in that space of possibilities.

CHAPTER 3

Literature on the Classification Problem

"those who ignore history are doomed to repeat the mistakes of the past.."

- George Santayana

An overview of the literature is important at this point, so we know what work has already been done on the classification problem. This allows us to incorporate ideas that worked and discard ideas that did not.

3.1 Supervised vs. Unsupervised

The literature for radar-based classifiers usually categorizes classification algorithms into two different types: supervised and unsupervised.

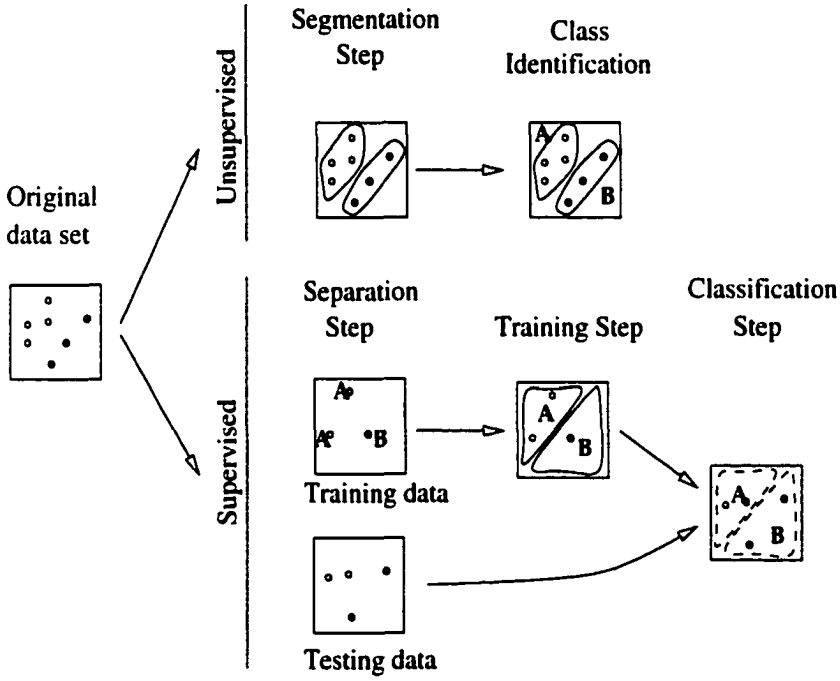


Figure 3.1: The difference between unsupervised and supervised classifiers.

An *unsupervised* classifier is given raw data, and the classification algorithm separates data into clusters, without any extra information. Each cluster must subse-

quently be identified as a class by hand.

A *supervised* classifier requires more preparation; it must be given a set of data with corresponding classes (usually called training data) with which it tries to 'learn' the character of that particular class. After this step, it uses what it learned and applies it to classifying any other data that it is given.

3.2 Standard Classification Techniques

We will now briefly describe a set of mathematical techniques that are frequently applied to the classification problem. This techniques described are extremely simple and very common, and can be found in almost any introductory textbook on classification such as [16]. They can be thought of as a collection of tools, which must be applied to each situation at hand. These tools can be combined in many different ways to perform classification, but they usually don't specify the details of their implementation.

An exhaustive list of existing tools in our tool-box would take too long, so we will only attempt to give the reader a flavor of some of the more common tools that we have at our disposal, and how they work.

3.2.1 Method of Principal Components

The method of principal components is a preprocessing step for multidimensional data, which is used to find new set of basis vectors that the data is projected into. The new basis is constructed so that correlation between the dimensions of the projected

data are minimized, and this can be used to reduce the dimensionality of the data set by discarding channels which are highly correlated.

Consider a situation where each data point is a realization of a multivariate random variable $\boldsymbol{X} = (X_1, X_2, \cdots)$. To preprocess the data using the method of principal components, we would compute coefficients C_j and project the data into a new basis by taking weighted sums of each measured data point, $Y_i = \sum_j C_j X_j$. From a statistical perspective, we have produced a new set of random variables that are linear combinations of the originals. We can call the new data $\boldsymbol{Y} = (Y_1, Y_2, \cdots)$.

3.2.2 Markov Random Field Models

Markov random field models are an important family of statistical representations that attempt to describe the texture and spatial variation in an image. The easiest way to understand MRF models is to extrapolate from Markov random processes.

Like any random process, a discrete Markov random process produces a string of random numbers $(x_1, x_2, x_3, ..., x_N, ...)$. A Markov random process is characterized by the fact that the statistics of the x_{N+1} value are determined completely by $(x_{N-a}, ...x_N)$ values. For a Markov random process, we can write,

$$P(x_{n+1}|x_n, x_{n-1}, \ldots, x_0) = P(x_{n+1}|x_n) \tag{3.1}$$

A Markov random field, used in the context of image processing, is closely related. It tells us that the statistical description of a pixel is conditionally dependent on the pixels immediately surrounding it. The further away from the pixel you get, the less the dependence becomes, until at some distance, the effect is essentially negligible.

This fits with the intuitive understanding that if you know that one pixel is grass, then it is likely that there is a field of grass surrounding it. This does not preclude the possibility of there being a fence there, but tells us that it is a less likely proposition.

These techniques are usually used by postulating or developing the probability density function of a pixel class conditioned on the classes of the pixels around it.

3.2.3 Maximum Likelihood Estimation

The Maximum Likelihood Estimator (MLE) is a statistical method of estimating the parameters of a pdf, given a set of observed random samples from that pdf.

Assume $\boldsymbol{X}$ is a p dimensional random variable which has a pdf $f(\boldsymbol{X} = \mathbf{x}|\boldsymbol{Z} = \boldsymbol{z})$. $\boldsymbol{z}$ is a q dimensional vector of parameters that are unknown in the pdf (such as the mean, the standard deviation, etc..). Assume we have r independent measurements of the random variable $\boldsymbol{X}$, call them $\boldsymbol{M} = (\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots \boldsymbol{m}_r)$.

The pdf for measuring this particular set of data is given by

$$g(\boldsymbol{M}|\boldsymbol{Z} = \boldsymbol{z}) = \prod_{i=1}^{r} f(\boldsymbol{X} = \boldsymbol{m}_i|\boldsymbol{Z} = \boldsymbol{z}) \tag{3.2}$$

We can estimate $\boldsymbol{Z}$ by forming the likelihood function $L(\boldsymbol{Z} = \boldsymbol{z}) = g(\boldsymbol{M}|\boldsymbol{Z} = \boldsymbol{z})$. Now solving $\frac{\partial}{\partial \boldsymbol{Z}} L(\boldsymbol{Z} = \boldsymbol{z}) = 0$ will give us the local extrema for the likelihood function. Finding the maximum of this function gives us the roots that are consistent with our data.

Maximum likelihood ratios of one pdf to another can be used when you have a set of data and you need to choose between two possible pdfs that may have generated it. The ratio then indicates which one is more likely to have generated the data.

3.2.4 Maximum *a posteriori* Estimation

The Maximum *A Posteriori* estimator (MAP) is a statistical method which we use to estimate the parameters in a pdf, given random samples from that pdf and a distribution of the parameters.

Assume $\boldsymbol{X}$ is a p dimensional random variable which has a pdf $f(\boldsymbol{X} = \mathbf{x}|\boldsymbol{Z} = \boldsymbol{z})$. $\boldsymbol{Z}$ is a q dimensional random vector of parameters that are unknown in the pdf. Also assume we know $f(\boldsymbol{Z} = \boldsymbol{z})$, the pdf of the parameters. Also assume we have r independent measurements of the random variable $\boldsymbol{X}$, call them $\boldsymbol{M} = (\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots \boldsymbol{m}_r)$.

As in the maximum likelihood estimation, we form the pdf for measuring this particular set of data

$$f(\boldsymbol{M}|\boldsymbol{Z} = \boldsymbol{z}) = \prod_{i=1}^{r} f(\boldsymbol{X} = \boldsymbol{m}_i|\boldsymbol{Z} = \boldsymbol{z}) \tag{3.3}$$

The major difference between the MAP estimator and ML estimator is that in the former, we assume we have the distribution of the parameters, $f(\boldsymbol{Z} = \boldsymbol{z})$.

Our goal now is to maximize $f(\boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{M})$ by finding the appropriate $\boldsymbol{z}$. The standard technique is to set the partial derivative to zero, $\frac{\partial}{\partial \boldsymbol{Z}} f(\boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{M}) = 0$, and solve for $\boldsymbol{z}$.

Bayes' rule tells us that $f(\boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{M}) = f(\boldsymbol{M}|\boldsymbol{Z} = \boldsymbol{z}) f(\boldsymbol{Z} = \boldsymbol{z}) / f(\boldsymbol{M})$. Since $f(\boldsymbol{M})$ does not vary with $\boldsymbol{Z}$, maximizing $f(\boldsymbol{Z} = \boldsymbol{z}|\boldsymbol{M})$ is the same as maximizing $f(\boldsymbol{M}|\boldsymbol{Z} = \boldsymbol{z}) f(\boldsymbol{Z} = \boldsymbol{z})$.

Thus, solving for the maximum a posteriori estimate is equivalent to solving $\frac{\partial}{\partial \boldsymbol{Z}} f(\boldsymbol{M}|\boldsymbol{Z} = \boldsymbol{z}) f(\boldsymbol{Z} = \boldsymbol{z}) = 0$.

3.2.5 Optimal Bayesian Classification

Optimal Bayesian classification is a technique that relies on a direct, simple application of Bayes' theorem, and a comparison of probabilities.

Bayes Theorem states:

$$P(A = a|B = b) = \frac{P(B = b|A = a)P(A = a)}{\sum_i P(B = b|A = i)P(A = i)} \tag{3.4}$$

If $B = b$ is short hand for the statement "Sensor B measured b" and $A = a$ is short hand for "the class A of the measurement is a", then knowing the probability of each class $P(B = b)$ and the distribution of the measurements in each class $P(B = b|A = a)$ allows us to solve for the probability that a measurement is in a particular class $P(A = a|B = b)$.

Having done this for each class, we can compare probabilities and choose the most probable class, given the data that we measured.

3.2.6 The Minimum Distance Classifier

The minimum distance classifier creates regions based on a set of region centers in the measurement space. The geometrical distance between a data point and each region center can be computed and compared, and the region center that the datum is closest to is the minimum distance.

For example, if $\boldsymbol{r}_i$ is the vector that describes region i, and $\mathbf{x}$ is a data point, then computing the distance is as simple as evaluating $|\boldsymbol{r}_i - \mathbf{x}|$.

3.2.7 Parallelepiped Classifier

Used only for supervised classification, the parallelepiped classifier is best thought of as distinguishing data on the basis of regions which are easily described by the union of a set of multidimensional rectangles.

Decision boundaries (or thresholds) must be specified. For example, if we had a data point $\mathbf{x} = (x_1, x_2)$ we can say it is class A if $l_1 < x_1 < u_l$ and $l_2 < x_2 < u_2$. These regions can be quite complicated, but it is then simple to use Boolean logic and compute whether a point is within those boundaries.

3.3 Recent Literature on Classification

Here we attempt to give a quick overview of important classification techniques in the literature. Unlike the previous section, where we described simple and very common techniques, these papers typically present details and implement the classification for a particular purpose. Many of these techniques are specifically tailored to classification using SAR data.

It should also be noted that some of these techniques can be used for supervised as well as unsupervised classification. For example, ISODATA was developed as a clustering technique for unsupervised classification, but given training data, it can be used to find region centers for a minimum distance classifier.

3.3.1 ISODATA

ISODATA was fist published by Ball, *et al.*, [2] and is a clustering technique which is typically used in unsupervised classification techniques. ISODATA stands for "Iterative Self-Organizing Data Analysis Techniques" with an "A" tacked on the end for good measure.

The idea behind ISODATA is that the data will somehow separate itself into clusters or groups of similar points, where each group could be a subset of a larger class. The algorithm attempts to mathematically determine which points belong together.

As an input, it takes a set of multidimensional data, and a series of initial cluster points. Each cluster point defines the centroid of some region which encloses some of the data. The regions are defined such that a point belongs in the region defined by the closest cluster point (i.e., a minimum distance metric).

Each iteration examines the data points in each region based on their standard deviation in each dimension; if it is too large, then the region splits into two regions. Pairs of regions are also examined, such that if two regions are close together and certain other criteria are met, they are combined and lumped into one. The geometrical average of the points in a region is found, and these average points are used as the new cluster centers. The new regions are found and the process repeats itself, eventually converging.

3.3.2 Entropy–Based Classification Scheme

Cloude, *et al.*, [6] developed a technique that uses the coherency matrix of SAR data to perform a supervised classification.

The coherency matrix is simply a recombination of terms in the covariance matrix of a multivariate data set. This scheme assumes that this matrix is diagonalizable and that the eigenvectors are of a particular form. Extracting a parameter (α) from the eigenvectors and plotting it against the entropy (H) of the eigenvalues, all the data can be represented on a two-dimensional plane. They then hypothesized relationships between regions on the α-H plane and electromagnetic scattering mechanisms. On the basis of these regions, they group data with similar scattering mechanisms (and hopefully geometries) together.

The technique has applied to various SAR images, but no numerical accuracies were given.

3.3.3 Iterated Conditional Modes Algorithm

The Iterated Conditional Modes algorithm (ICM) begins by assuming a Markov random field model $p_i(x_i|\hat{x}_N)$, where x_i is the classified value of the pixel of interest and $\hat{x}_N$ is an estimate of the classified values of pixels in the neighborhood N of i.

It then defines y to be the sensor measurements of all the pixels in the image, and $\hat{x}_S$ to be the estimate of the classified values in the set S, which is the set of all the pixels in the image except the current pixel of interest i. Using Bayes' Theorem we

can write

$$P(x_i|y, \hat{x}_S) \propto f(y_i|x_i)p_i(x_i|\hat{x}_N) \tag{3.5}$$

For each pixel, we can iterate by maximizing $P(x_i|y, \hat{x}_S)$ with respect to x_i. In doing this, we find an estimate of the classification value for that iteration ($\hat{x}_i$). When the rest of the pixels are updated, our knowledge $\hat{x}_S$ changes and a new iteration starts, since we can again maximize $P(x_i|y, \hat{x}_S)$ for pixel i to find a new $\hat{x}_i$.

After sufficient iterations, the classification values converge, giving us our final image.

3.3.4 MCLUST

Fraley and Raftery [10] developed an unsupervised classification algorithm that models a set of data by assuming the underlying distribution was a mixture of multivariate Gaussian distributions. Then they used the Bayesian Information Criterion to choose between models with different number of clusters and different cluster densities. The clustering itself was done using a root finding technique, in this case the estimation–maximization algorithm.

A good source of information which is not found in the usual journals is at the following web site:

`http://www.stat.washington.edu/fraley/mclust_home.html`

This method was not applied to SAR images, and no numerical accuracies were given.

3.3.5 A Markov Random Field Model

Solberg *et al.*, [31] developed a model based on Markov Random Fields which was used to fuse or combine information provided by different sources of data. After the fusion, both maximum *a posteriori* and the iterated condition modes algorithms were applied to do the actual classification.

Data were taken from three different days, and five different level one classes were chosen: water, urban, forest, plowed and unplowed. Accuracies ranged from 80 to 100%.

3.3.6 Segmentation using the Covariance Matrix

Rignot, *et al.*, [27] used the ISODATA clustering algorithm to cluster the data initially, and followed that by clustering using a fuzzy c-means algorithm. The fuzziness in the second clustering comes from the distance metric, which takes the distance between the possible cluster center and the data, but multiplies it by a probability computed assuming the data can be a mixture of classes. Once the cluster centers are determined, the MAP classifier is used to classify the entire image.

No numerical accuracies were given.

3.3.7 Classification based on the Wishart Distribution

Lee *et al.*, [20] used the Wishart distribution (the pdf of a complex multivariate Gaussian) to create a distance measure. They then used this distance measure to create unsupervised and supervised classification techniques, by combining it with the

ISODATA clustering technique, and the maximum likelihood technique, respectively.

This unsupervised clustering technique was applied to sea ice and separated four classes (open water, first year ice, multi year ice, and ridge) with between 80% and 100% accuracy. The supervised classification was applied to an image of San Francisco and separated five level-one classes (ocean, park, city, and quasi-natural) with between 60% and 90% accuracy.

As with other efforts in the literature, these classification results are estimted, and not based on actual ground-truth observations. The real-world performance might actually be far worse.

3.3.8 Knowledge-Based Hierarchical Classifier

Pierce *et al.*, [25] selected two measurements for each class of interest, and plotted each point in a cartesian coordinate system. For each class, they drew piecewise linear boundaries that seperated that class from all the others. Then, it was easy to identify whether a particular data point was within the specified boundary (belongs to the class) or not within the boundary (belongs to some other class).

With four level-one classes, accuracies above 90% were achieved.

3.3.9 Weighted Clustering Methods

Jeon, *et al.*, [17] trained a classifier on samples of one class for the purpose of separating this class from everything else. This is a partially supervised classifier in which they assume that the pdf of the class of interest in measurement space is

known.

This pdf is used along with the data to estimate a weighting for each data point, and unsupervised clustering is performed on the weighted data. A conventional supervised clustering technique is then used, such as ISODATA or the expectation maximization algorithm, to further refine the classes.

With four short-vegetation classes, (corn, soybeans, wheat, and alfalfa) using LANDSAT data, this technique produced errors that were as high as 16% for a particular class. They measured commission errors between 2 and 16%, and omission errors between 3 and 16%.

Of all the literature discussed so far, the study by Jeon, *et al.* most closely resembles the objectives of our investigation, and is one of the few that has a very large data set. Like many others in the literature, their data set does not span multiple dates. They also make the common assumption that the probability density function of the classes are Gaussian.

3.3.10 Neural Networks

Ito and Omatu [13] used the concept of competitive neural networks to develop a classifier. Each output is a summation of terms, with one term for each input in the measurement's dimension. Each input-output combination has a unique connection weight which is determined during training.

For each measurement, the outputs are compared, and the winning output determines how that datum is classified. The average accuracy for five level-one classes

(factory, golf course, vegetation, urban, and water) was between 80 and 100%.

3.3.11 Semivariogram Texture Measure

Miranda *et al.*, [23], computed the semivariogram texture of various images. The semivariogram texture is a metric that they define and compute to represent the texture, or spatial variations of the image due to the vegetation geometry. They demonstrated that this measure of texture is extremely important in classification.

On the basis of this feature and the digital number of the SAR data alone, they were able to separate four level-one classes (water, open vegetation, dense vegetation, and flooded vegetation) with accuracies between 58 and 68%.

3.3.12 Classifiers Based on Polarimetric Filtering

Nagai, *et al.*, [24], used pixels collected from areas of interest to find the polarization signatures of each of the classes. The contrast between classes was computed, and the polarization producing the highest contrast was chosen as the optimum polarization for separation. Polarimetric synthesis was done on the image (this is the filtering step) and then a maximum likelihood classification scheme was run on the result.

For three level-one vegetation classes (vegetation, farm, and water), accuracies ranged between 80% and 90%, while a residential class was classified with an accuracy of 47%.

Sarabandi and Li [30] used a Genetic Algorithm to find the optimum polarization

which could be used to separate two classes. To classify each pair of classes, polarimetric synthesis was done on the data and a simple thresholding was performed.

The algorithm was used to classify different road conditions such as concrete, wet concrete, icy concrete.

Swartz *et al.* [36] also tried to find the optimal polarization for separating two different types of scatterers, instead by developing a matched filter which found a contrast ratio.

3.4 Observations Regarding the Literature

Our aim, now, is to learn from the literature, and we can do this by adopting successful ideas and addressing issues that caused problems in the past.

Some successful ideas, such as neural networks and optimal polarization are excellent for other types of classification problems, but do not work well on the problem at hand. Both techniques have difficulties because they do not adequately take into account the variation in measurements that we see over time for a single vegetation type. One promising way to do this is to use the correlations between different channels of measurement. In the case of the optimal polarization technique, it discards many of the multiple dimensions of the measurement, ignoring the correlations completely.

Other techniques like the semivariogram texture measure and Markov random field models are demonstrably useful, because they use the spatial information available in an image to deal with the the classification problem. Our investigation focuses only on the electromagnetic scattering characteristics of each pixel in isolation. For this

reason, we do not try to exploit information about the spatial correlation information, of the data. These models would enhance any classification technique we develop, and combining them could be a topic of future investigations.

Supervised methods in the literature are useful because they have the capability of producing fully automated classifiers. Unsupervised methods require human intervention after the clustering phase, and are less desirable. In addition, methods like ISODATA have poor accuracy when applied to classification of short vegetation, usually because they incorrectly group the clusters together, erroneously lumping one class within another.

We can also note areas that have not been explored in the literature. These are aspects of the classification problem that we would like to address.

1. Although many classification techniques exist to deal with multivariate data, the SAR literature does not typically apply these techniques to their fullest potential.

   Frequently, the representation or model development only goes up to characterizing the data up to its second order statistics (e.g. [20]). Although radar measurements of vegetation have a Gaussian character if we take repeated measurements under fixed conditions, this is not the case for a typical vegetation type, whose height, biomass, moisture, and other biophysical parameters change from field to field, as well as varying temporally.

   In other cases, the sensor is not fully polarimetric, and the potential multivariate character of the data is not fully realized (e.g. [23]). Many times, SAR data

is polarimetrically filtered, thus reducing multivariate data to a single scalar power value.

We would like to develop a method that has potential to provide information about electromagnetic scattering statistics, while still performing well for classification.

2. We would like to be able to represent the temporal variation of the radar measurments of vegetation. Few classifiers treat this important aspect explicitly.

3. We would like our classifier to be sensitive to fine geometrical differences, and be sophisticated enough to perform level two classification. Many classifiers identify classes with gross geometrical differences, such as short vegetation and trees. Techniques that do deal with finer distinctions typically have poor accuracies.

4. We would like our classifier to be robust. This simply means that we must have enough data to do the classification properly, and accurate ground-truth information to evaluate the end result.

   Frequently, classifiers presented in the literature are developed with very small data sets. As a consequence of this, these classifiers sometimes use the same data set for training and testing the technique. This leads to techniques that work well on a single image, but do not work when applied to other measurements with different conditions.

   Other times, there are no measurements of what is on the ground to compare

results with, so it is unclear how accurate the classification really is.

CHAPTER 4

Developing Classification Algorithms for Short Vegetation

We have discussed at length the system we are using, the statistics that affect it, the data acquisition and processing, existing techniques in the literature, as well as a systems level perspective of the measurement process. How do we actually combine this information into a coherent system that will quickly and accurately go about classifying this data?

Armed with what we now know, we will describe a series of different classifiers, starting with the Hierarchical classifier in [25]. Each subsequent classifier builds upon the previous ones, inheriting and improving on their strengths and addressing their weaknesses.

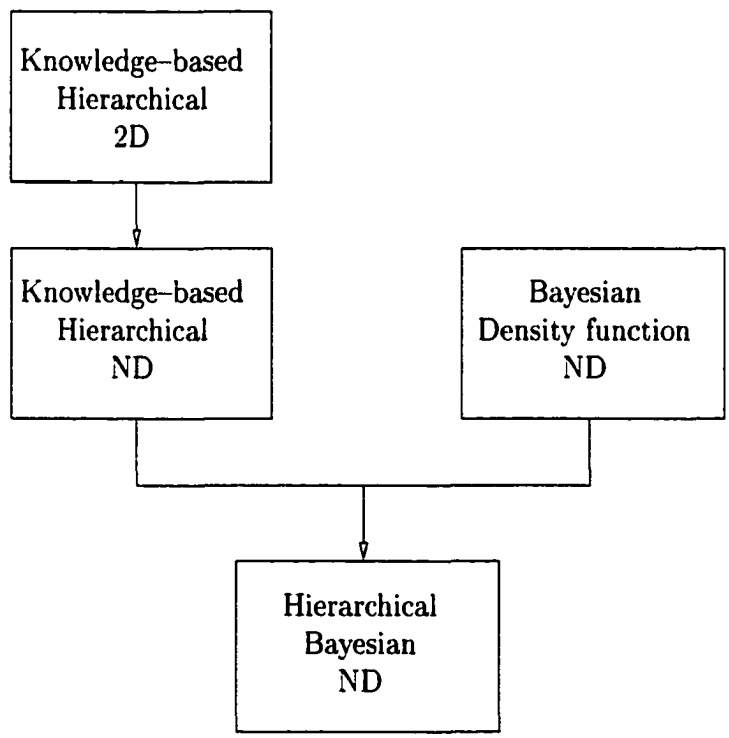


Figure 4.1: A map of the different classification techniques we will discuss, where each builds on the best traits of the previous one.

Figure 4.1 is a roadmap of the different classifiers we will discuss and how they

are related to each other. The box on the upper left represents the Hierarchical two-dimensional classifier in [25], and below it is an improved N-dimensional version.

The box on the right is a Bayesian technique in multiple dimensions that uses smooth, analytic probability density functions. This technique has a sound statistical basis, but there are some impediments to using it directly, most notably the lack of a computationally efficient, multidimensional density estimation technique, and the relatively small amounts of training data that we have to characterize the statistics of our data.

We will end this chapter with a discussion of the Bayesian technique and proceed into the realm of density estimation in the next chapter. In Chapter 6, we will demonstrate how the density estimation technique can be used with the Bayesian technique and applied to practical classification problems.

4.1 A Knowledge-Based, Hierarchical Classifier in Two Dimensions

We begin with the box in the upper left of Figure 4.1. This Knowledge-Based, Hierarchical classification technique, developed by Pierce *et al.* in [25] is a starting point of the research done for this dissertation.

This classification technique was used to produce a level-one classifier, which seperated classes such as trees, water, short vegetation and tall vegetation.

After processing and organizing our training data, we have a set of data measured

from each different class. Each of these data sets can be plotted as a cloud of points (with each plotted point being a measurement of the vector M in Figure 2.4) in our multidimensional measurement space. Setting aside difficulties of dimension, it seems logical that if we have chosen our classes correctly, the cloud for each class might be somehow disjoint from the clouds of other classes.

Our first instinct might be to draw boundaries between the clouds which would allow us to separate each cloud from the rest. The difficulty occurs because we cannot visualize the data plotted in this multidimensional space.

For each stage, the set of data that we have is projected into two dimensions, and plotted. Piecewise linear boundaries are drawn between the class of interest and the rest of the classes, and the remaining data is projected into another two dimensions of interest. Figure 4.2 illustrates the boundaries that were drawn in the training process.

This method is hierarchical because it works in stages. The first stage distinguishes class A from all the rest, and the second stage separates class B from the rest, and so on. If plotted graphically, the hierarchy of decisions that make up the classifier look like a branching tree.

	Classified As		
True Class	Tall Veg	Short Veg	Bare
Tall Veg	100.0	0.0	0.0
Short Veg	0.0	99.12	0.88
Bare	0.0	2.06	97.94

Table 4.1: Level-one classification accuracies produced by the Knowledge-Based Hierarchical technique.

In the case of this level-one classifier, this technique does its job very well. The accuracies in Table 4.1 demonstrate excellent performance, but applying this classi-

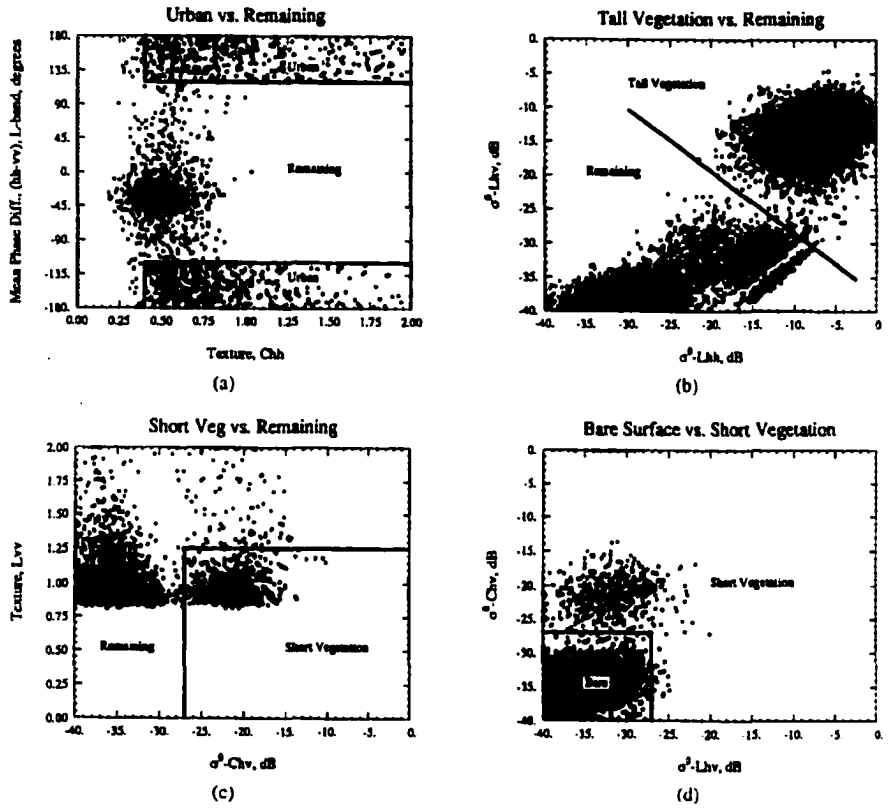


Figure 4.2: Piecewise linear boundaries used for separation of four level-one classes.

fication technique to level-two classes, we discover that there are unacceptably high levels of error in level-two classification.

4.2 A Knowledge-Based, Hierarchical Classifier in Multiple Dimensions

To improve on the Knowledge-Based classifier described in the last section, we would like to allow each decision boundary to better exploit the multiple dimensions we have in our measurements. Currently, each decision boundary only uses two dimensions of information in the data.

When we plot a two-dimensional projection of our measurement space and draw boundaries, we can use the correlations between those two dimensions to help us. Our problem might be solved by looking at the correlation between more than two dimensions of data. The main problem with the hierarchical classifier is that plotting the points in two dimensions does not fully exploit the relationships between the rest of the dimensions of the data. The method only uses that two dimensional projection and its respective correlations.

Unfortunately, we simply cannot geometrically 'see' the correlations in higher dimensional spaces, and it is very difficult to represent them on paper.

It is, however, possible to create a higher dimensional analogy to the knowledge-based method in the previous section, and this is the second tier box on the left of our roadmap in Figure 4.1. This is also a knowledge-based, hierarchical classification technique, but has a few improvements, allowing each decision rule to work with a total of more than two dimensions.

A quick example at this point is instructive. In our particular instance, we are finding the decision rules to separate the corn crop measurements from the rest of the measured data from other vegetation types. Each measured data point in each frequency band was represented by a co-polarized α and the three different polarizations of backscattering coefficients (σ^0_{HH}, σ^0_{VV}, σ^0_{HV}).[1] With two different freuency bands (L and C band), this gave us a total of eight dimensions for each data point.

As before, we project our eight dimensional data into two dimensions, and plot

[1]It is also interesting to note that in producing this example, we never used the co-polarized phase statistic ζ. This is because ζ does not appear to add any valuable information to the classification procedure. Removing it has absolutely no effect on the classification accuracies.

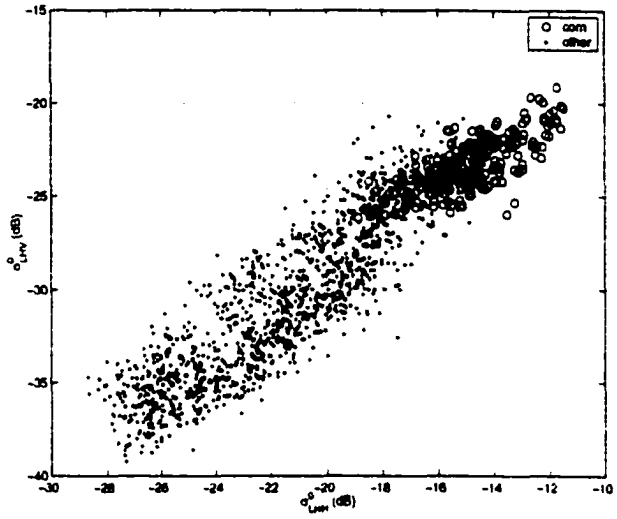


Figure 4.3: The first step of a multidimensional hierarchical decision rule is that we project our measured radar data into two of its dimensions.

it, as in Figure 4.3. We draw our boundaries so that on one side of the boundary, all of the data is definitely not corn. The data points that remain on the other side of the boundary are a mixture of measurements from our corn, as well as data from the other vegetation classes. Figure 4.4 shows the boundaries we have drawn, and Figure 4.5 shows us discarding the measured data points to the left of the boundary, which we have identified as not-corn.

The measured data that remains in Figure 4.5 is then reprojected onto two different dimensions, and the process is repeated. Reprojecting our data into different pairs of dimensions is akin to looking at the multidimensional clouds of data from different angles. Each additional view we get can give us new information.

Figure 4.6 shows the results of reprojecting the radar measurements from Figure 4.5 onto two new dimensions, and drawing boundaries. In Figure 4.7 the not-corn side of the boundary is discarded.

What remains after we are done is the set of points that are always in the region

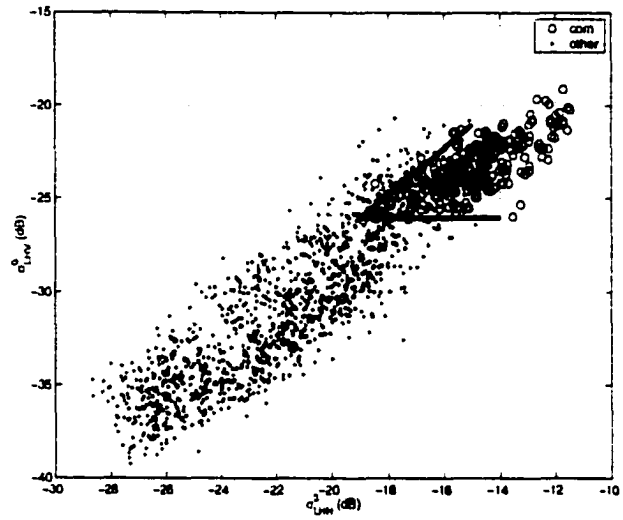


Figure 4.4: The second step of a multidimensional hierarchical decision rule is that we draw boundaries separating possibly-corn measuremnts from non-corn measurements.

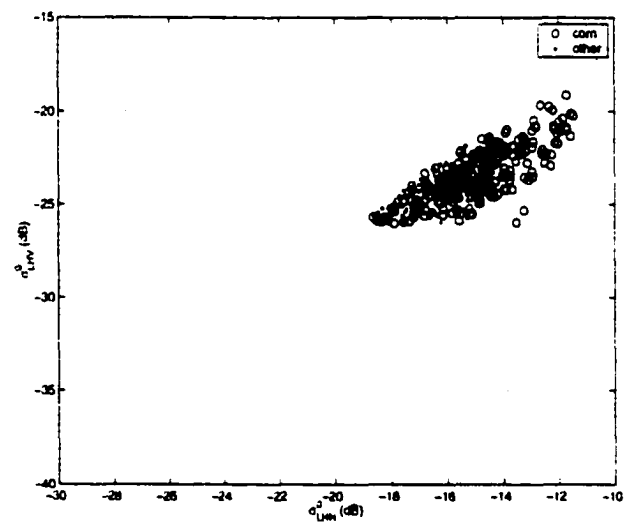


Figure 4.5: The third step of a multidimensional hierarchical decision rule is that we discard measured data that has been identified as non-corn. Note that the remaining data still has data measured from other classes mixed in with the corn data.

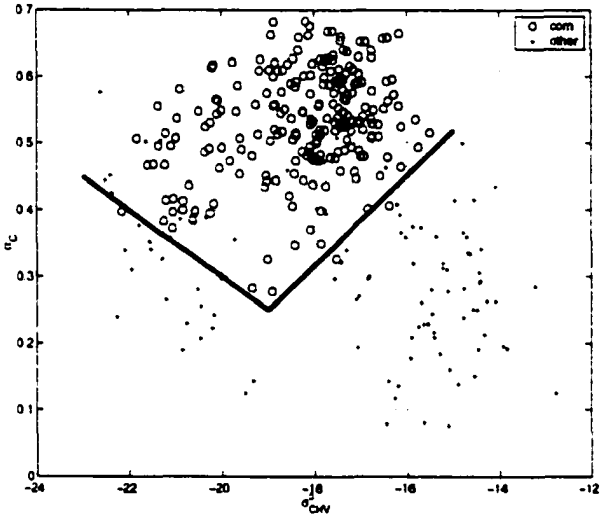


Figure 4.6: The fourth step of a multidimensional hierarchical decision rule is that we reproject measured data that has been identified as possibly-corn onto a new set of dimensions. This exposes non-corn regions, and allows us to draw boundaries.

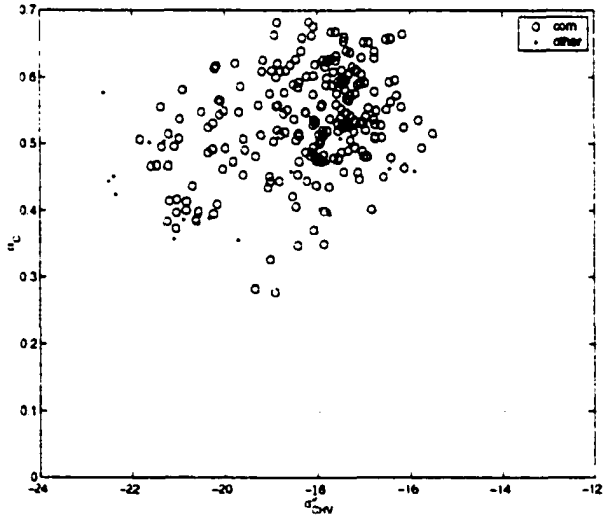


Figure 4.7: The fifth step of a multidimensional hierarchical decision rule is that we again discard measurements in the non-corn region of the plot. This process can be repeated with as meny projections as are useful.

of our higher dimensional measurement space that contains our class of interest (in this case corn). We have, piece by piece, defined a multi-dimensional boundary. This technique was demonstrated by Kouskoulas, *et al.* in [19].

In the preceeding paragraphs, we have described the process of drawing boundaries and how this allows us to separate corn from other classes. This process is repeated on the non-corn data that remains, for each additional vegetation class we wish to identify.

A detailed, step-by-step graphical example of applying this technique to short vegetation is presented in Figures 4.8-4.18. Note that the superscripts distinguish between early and late wheat and short and tall alfalfa.

At the end of our example, and during the process of classification, we present the results in the form of what we call confusion matrices. Each entry in the confusion matrix is a percentage, indicating what percent of the data in the class labeled on the left side of the row was classified with the label in the column above. This is the convention we use for all of the confusion matrices in this text.

Table 4.2 gives the final results in confusion table form for the data that we developed the decision rules on, which we call our training data set. It is important to note that any time we develop a classifier, we must test it on an independent data set. Table 4.3 gives a confusion table of the results of the classifier tested on an independent data set.

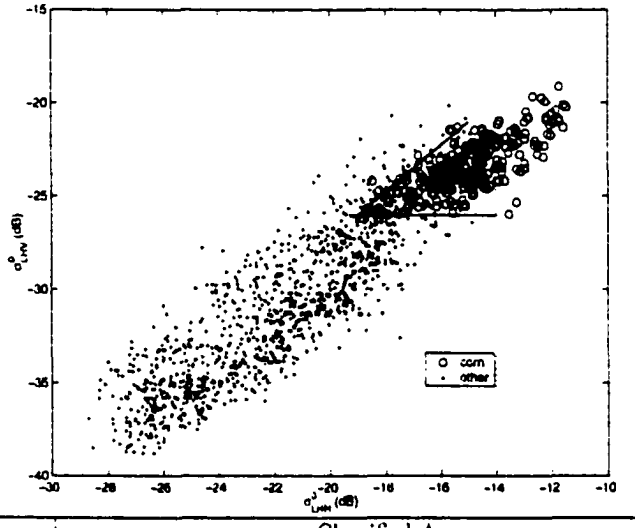
4.2.1 Graphical representation of decision rules

	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$	91.0	1.4	0.6	2.4	0.0	4.6
Alfalfa$^{(T)}$	81.1	7.2	0.0	0.0	1.7	10.0
Bare	2.1	85.0	4.9	0.0	6.2	1.8
Corn	0.4	3.9	95.3	0.0	0.4	0.0
Soy	7.4	0.0	0.0	89.4	0.0	3.2
Wheat$^{(E)}$	0.5	2.4	5.3	0.0	88.2	3.6
Wheat$^{(L)}$	9.1	1.9	0.0	1.9	1.9	85.2

Table 4.2: Accuracy of Hierarchical classification using training data.

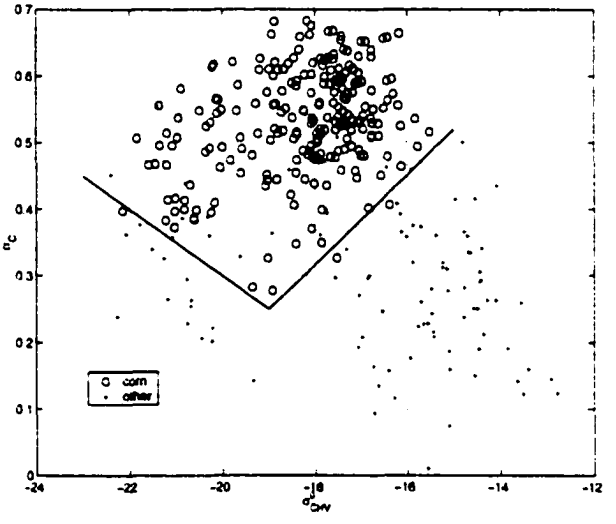
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$	88.4	1.7	2.2	3.2	0.2	4.3
Alfalfa$^{(T)}$	81.4	8.3	0.0	0.3	4.2	5.8
Bare	6.2	80.9	4.5	0.0	6.9	1.5
Corn	0.0	3.8	94.8	0.0	1.3	0.2
Soy	12.0	0.0	0.0	86.5	0.0	1.5
Wheat$^{(E)}$	0.7	3.0	3.0	0.0	86.3	7.0
Wheat$^{(L)}$	14.5	2.5	0.4	1.3	2.0	79.3

Table 4.3: Accuracy of Hierarchical classification using independent testing data.



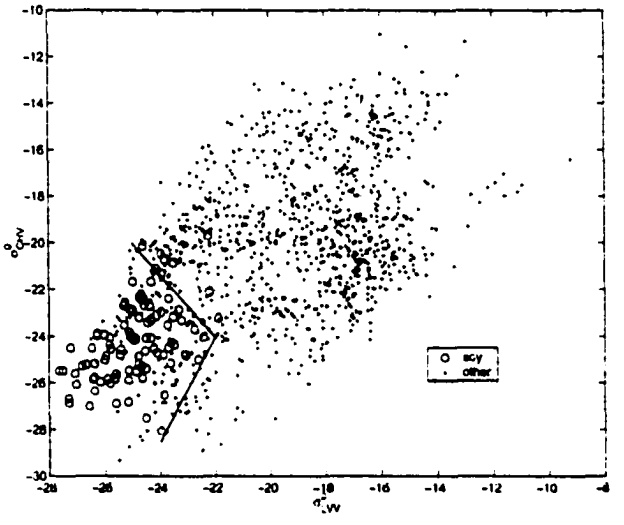
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$			1.4			
Alfalfa$^{(T)}$			8.3			
Bare			26.0			
Corn			96.6			
Soy			0.0			
Wheat$^{(E)}$			14.2			
Wheat$^{(L)}$			0.0			

Figure 4.8: Separating corn, projection A



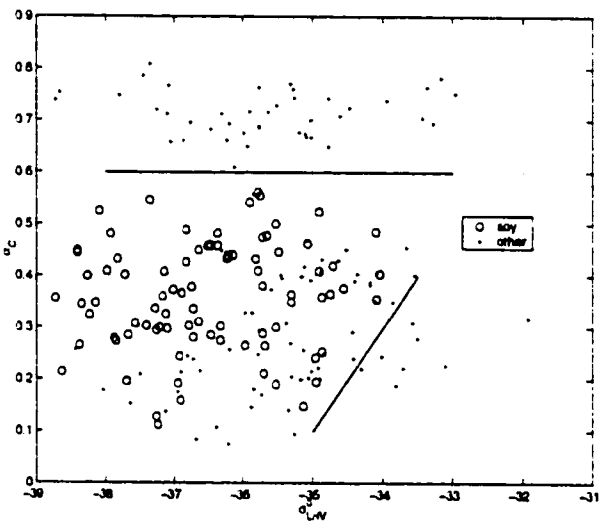
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$			0.6			
Alfalfa$^{(T)}$			0.0			
Bare			4.9			
Corn			95.3			
Soy			0.0			
Wheat$^{(E)}$			5.3			
Wheat$^{(L)}$			0.0			

Figure 4.9: Separating corn, projection B



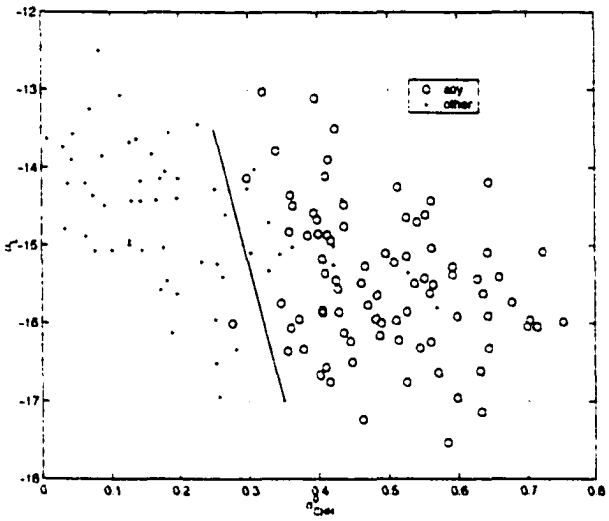
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$			0.6	25.8		
Alfalfa$^{(T)}$			0.0	10.0		
Bare			4.9	0.0		
Corn			95.3	0.0		
Soy			0.0	90.4		
Wheat$^{(E)}$			5.3	0.0		
Wheat$^{(L)}$			0.0	8.6		

Figure 4.10: Separating soybeans, projection C



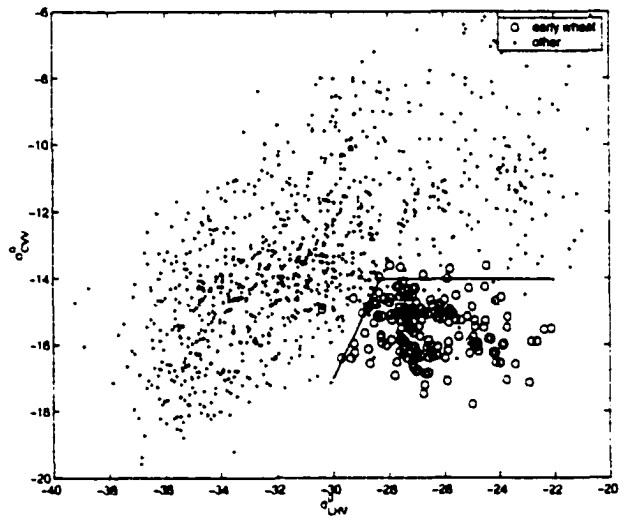
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$			0.6	12.7		
Alfalfa$^{(T)}$			0.0	0.0		
Bare			4.9	0.0		
Corn			95.3	0.0		
Soy			0.0	90.4		
Wheat$^{(E)}$			5.3	0.0		
Wheat$^{(L)}$			0.0	6.7		

Figure 4.11: Separating soybeans, projection D



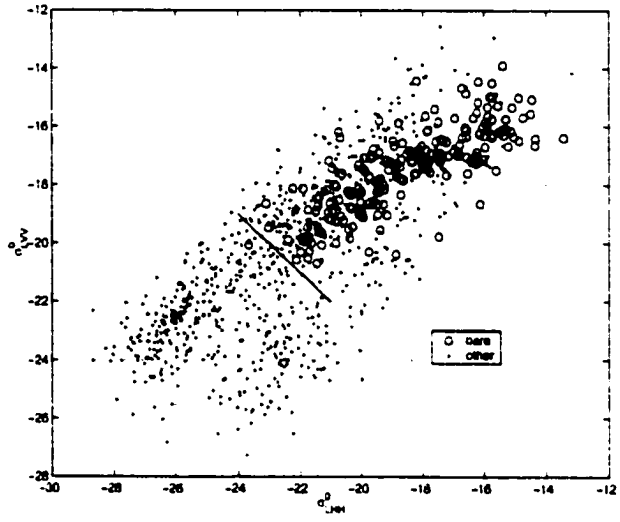
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$			0.6	2.4		
Alfalfa$^{(T)}$			0.0	0.0		
Bare			4.9	0.0		
Corn			95.3	0.0		
Soy			0.0	89.4		
Wheat$^{(E)}$			5.3	0.0		
Wheat$^{(L)}$			0.0	1.9		

Figure 4.12: Separating soybeans, projection E



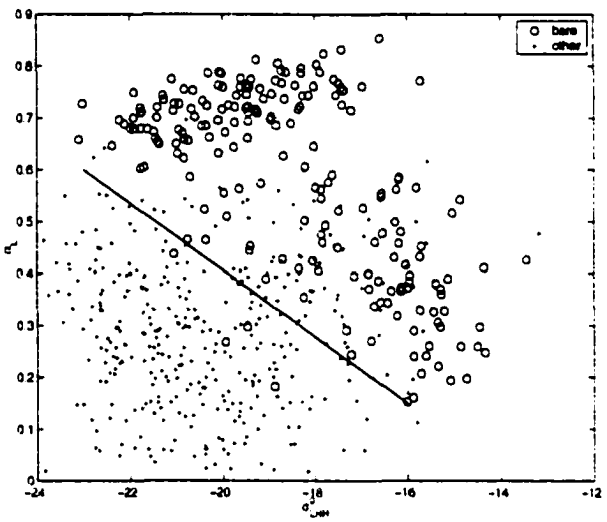
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$			0.6	2.4	0.0	
Alfalfa$^{(T)}$			0.0	0.0	1.7	
Bare			4.9	0.0	6.2	
Corn			95.3	0.0	0.4	
Soy			0.0	89.4	0.0	
Wheat$^{(E)}$			5.3	0.0	88.2	
Wheat$^{(L)}$			0.0	1.9	1.9	

Figure 4.13: Separating wheat, projection F



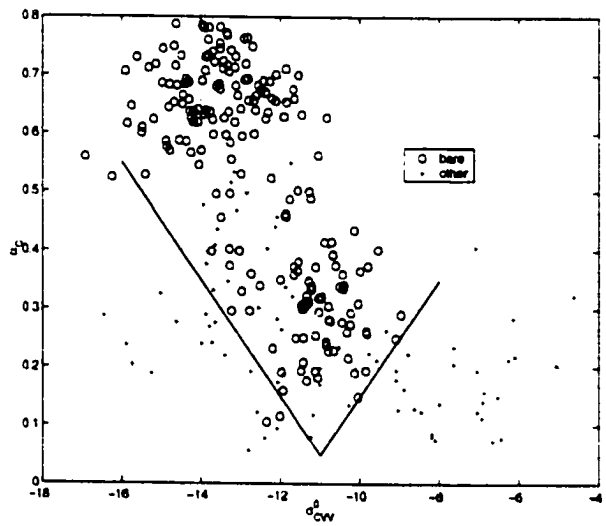
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$		25.2	0.6	2.4	0.0	
Alfalfa$^{(T)}$		56.1	0.0	0.0	1.7	
Bare		88.6	4.9	0.0	6.2	
Corn		4.3	95.3	0.0	0.4	
Soy		0.0	0.0	89.4	0.0	
Wheat$^{(E)}$		6.5	5.3	0.0	88.2	
Wheat$^{(L)}$		67.9	0.0	1.9	1.9	

Figure 4.14: Separating bare soil, projection G



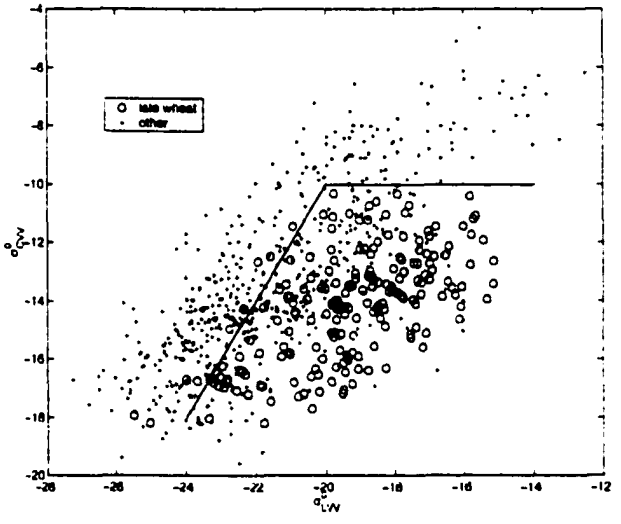
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$		6.5	0.6	2.4	0.0	
Alfalfa$^{(T)}$		13.9	0.0	0.0	1.7	
Bare		86.8	4.9	0.0	6.2	
Corn		4.3	95.3	0.0	0.4	
Soy		0.0	0.0	89.4	0.0	
Wheat$^{(E)}$		4.1	5.3	0.0	88.2	
Wheat$^{(L)}$		6.2	0.0	1.9	1.9	

Figure 4.15: Separating bare soil, projection H



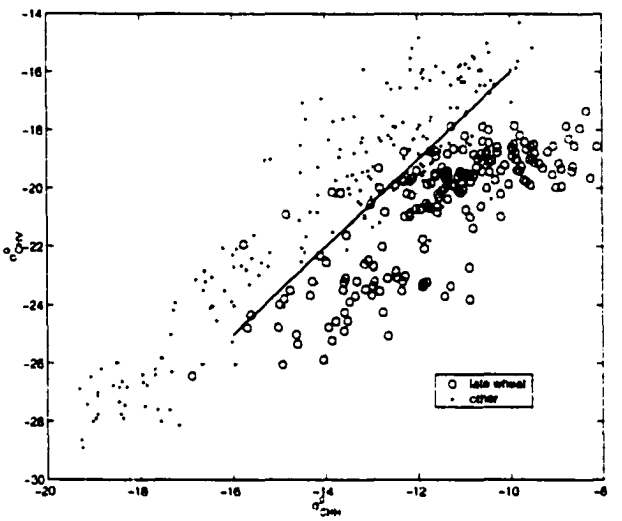
	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$		1.4	0.6	2.4	0.0	
Alfalfa$^{(T)}$		7.2	0.0	0.0	1.7	
Bare		85.0	4.9	0.0	6.2	
Corn		3.9	95.3	0.0	0.4	
Soy		0.0	0.0	89.4	0.0	
Wheat$^{(E)}$		2.4	5.3	0.0	88.2	
Wheat$^{(L)}$		1.9	0.0	1.9	1.9	

Figure 4.16: Separating bare soil, projection I



	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$		1.4	0.6	2.4	0.0	29.0
Alfalfa$^{(T)}$		7.2	0.0	0.0	1.7	49.4
Bare		85.0	4.9	0.0	6.2	3.9
Corn		3.9	95.3	0.0	0.4	0.0
Soy		0.0	0.0	89.4	0.0	3.2
Wheat$^{(E)}$		2.4	5.3	0.0	88.2	4.1
Wheat$^{(L)}$		1.9	0.0	1.9	1.9	90.0

Figure 4.17: Separating wheat, projection J



	Classified As					
True Class	Alfalfa	Bare Soil	Corn	Soybeans	Wheat$^{(E)}$	Wheat$^{(L)}$
Alfalfa$^{(S)}$	91.0	1.4	0.6	2.4	0.0	4.6
Alfalfa$^{(T)}$	81.1	7.2	0.0	0.0	1.7	10.0
Bare	2.1	85.0	4.9	0.0	6.2	1.8
Corn	0.4	3.9	95.3	0.0	0.4	0.0
Soy	7.4	0.0	0.0	89.4	0.0	3.2
Wheat$^{(E)}$	0.5	2.4	5.3	0.0	88.2	3.6
Wheat$^{(L)}$	9.1	1.9	0.0	1.9	1.9	85.2

Figure 4.18: Separating wheat, projection K

4.3 Optimal Bayesian Classification

Until now in this development, we have been dealing with sharply delineated thresholds in multiple dimensions. Although these boundaries give reasonably good results, various problems remain with this treatment of classification.

Firstly, our boundaries do not describe closed regions. When adding the ability to distinguish another target, we may have to retrain the classifier completely, as opposed to simply training it only for the new class. Secondly, our boundaries are sharply defined, and when classes overlap, the overlap is misclassified. Thirdly, human intervention is necessary to create the boundaries; the training stage is not automatic, and requires a great deal of labor.

To solve these problems, and to provide a rigorous mathematical framework for subsequent investigation, we explored the idea of representing each class of data as a multivariate probability density function.[2] A multivariate density describes a closed volume in multiple dimensions, and is a smoother representation of the boundaries we desire. We can think of a three dimensional density as a nested set of equiprobable surfaces. These surfaces define our boundaries, and the notion of probability density functions generalizes to higher dimensions easily. Use of the probability density function allows us to use Bayesian and other statistical methods for optimal classification of overlapping regions, which would solve the problem of the boundaries being too sharply defined.

The beauty of the representation as a pdf is that it is stunningly simple, and

[2]We will use the phrases "pdf," and "density" interchangably throughout the text to refer to probability density functions.

completely encompasses all the statistical correlations between classes.

Let us be precise about our intentions. We would like to use optimal Bayesian Classification to solve our classification problem, and identify different types of short vegetation.

Optimal Bayesian classification is usually formulated thus:

$$P(C = c|\boldsymbol{M} = \boldsymbol{m}) = \frac{P(\boldsymbol{M} = \boldsymbol{m}|C = c)P(C = c)}{\sum_a P(\boldsymbol{M} = \boldsymbol{m}|C = a)P(C = a)} \tag{4.1}$$

where $P(C = c|\boldsymbol{M} = \boldsymbol{m})$ is the probability that our measurement is a certain class 'c' if its value is $\boldsymbol{m}$. We can compute this probability if we know $P(\boldsymbol{M} = \boldsymbol{m}|C = c)$, the probability that we measure $\boldsymbol{m}$ if our class is 'c.'[3]

In terms of the definitions in Section 2.2.3, $P(\boldsymbol{M} = \boldsymbol{m}|C = c) = f_c(\boldsymbol{M} = \boldsymbol{m})$. With knowledge of this density, we can simply use optimal Bayesian classification.

Our first attempts to estimate $f_c(\boldsymbol{M} = \boldsymbol{m})$ were to compute a histogram, and simply use that representation. However, using a histogram as an estimate of our density is not particularly efficient, and needs a tremendous amount of data to be accurate. Secondly, with a limited set of data, the estimate is not only inaccurate, but also has a very jagged shape. The corners of the histogram cubes are likely to overlap even if the underlying densities do not. Thus, we would like to take our data and compute the smooth, analytical probability density function that produced it.

The optimal solution to the general problem of how to estimate an analytical density function based on a set of data has never been solved in generality.

Researchers have written books on density estimation, and many have developed

[3]We must also assume that we have no additional knowledge about the frequency of the classes (with N classes, that would mean that $P(C = a) = \frac{1}{N}$)

methods to estimate densities for specific types of problems, but all techniques examined have one or more of the following flaws:

- The technique is parametric, and assumes that the density is of a specific type. The assumption that the density is Gaussian is an example of this.
- The density estimation technique is unable to handle multivariate densities.
- The density estimation technique is not computationally efficient, and takes hours[4] to compute and manipulate densities.
- The density estimation technique has an ad hoc flavor, and no theoretical basis.

Thus, we would like to use a technique that avoids all of these flaws. It must have the ability to make good estimates based on a very limited set of data. We also want it to be able to produce higher dimensional densities that are computationally efficient to manipulate, as well as accurate when compared to competing techniques.

Once we can do the density estimate, we can classify into classes $A, B, C, \ldots$, by estimating $f_A(\boldsymbol{M} = \boldsymbol{m}), f_B(\boldsymbol{M} = \boldsymbol{m}), f_B(\boldsymbol{M} = \boldsymbol{m}), \ldots$ and applying them to our problem using optimal Bayesian classification.

[4]Using the computational power available in the University of Michigan Radiation Lab during the summer of 2000.

CHAPTER 5

A Computationally Efficient, Multivariate Maximum Entropy Density Estimation Technique

We now change gears and discuss the topic of density estimation. This chapter deliberately does not make explicit reference to the classification problem we have at hand, but the reader should keep it in mind. By developing the technique for the more general case of any multidimensional data set, we hope that it will be useful in other endeavors requiring density estimation.

5.1 Introduction

A data set is a collection of vectors, with each vector representing one data point, or a single measurement of interest. If the vector that represents a data point has more than one dimension, then the data set and data points are multidimensional, or multivariate. A biologist in Africa who records the height and weight of baboons born into a particular group is collecting a data set. So is a civil engineer who measures the x, y, and z displacement of a point on a concrete structure undergoing earthquake testing.

All areas of science work with data sets, and there are many occasions when we want to collect like data together and develop some sort of statistical characterization of that particular set of measurments.

Density estimation is the process of taking a set of multidimensional data that was produced by a particular random process, and finding the probability density function (pdf) that most likely produced it. The probability density function, in most cases, provides a complete description of the statistical character of the data set under consideration. This chapter proposes a very general approach for performing density

estimation with multidimensional data sets using an accurate and computationally efficient method.

5.1.1 Overview

Why density estimation is difficult

The data sets we are interested in are the ones that have an element of randomness to them, because otherwise, there would be no point in doing density estimation. Because of the inherently stochastic nature of the sort of data set we are dealing with, one quickly realizes that there is always some probability that a given density function could theoretically produce a particular set of data[1]. That is to say, given a set of data A, there is some probability that a χ^2 density produced it, but there is also some other probability level that an exponential density produced it instead.

As a consequence, we can never be entirely sure that we are correct in our density estimate. The best we can hope to do is to make statements like, "This data set is most likely to have come from a density that looks like this."

After all, it just might be that an exponential density produces a set of samples that are very uncharacteristic of it, and are closer to what is usually produced by a χ^2 density. So in the absence of other information, we sinply choose the guess that is most likely to be the correct one.

[1] Assuming that the data set and the pdf have the same domains.

Methodology

So how can we go about making a good guess about the underlying density that produced a given set of data?

First, we will need to use the data to guide us and constrain our search within the space of probability density functions. There are various simple ways to extract information from a set of data points.

- We can compute estimates of the joint moments. If a single data point is a realization of the random variables $X_1, X_2, \ldots, X_d$, we can estimate

$$E[X_1^{m_1} X_2^{m_2} X_3^{m_3} \cdots X_d^{m_d}] \tag{5.1}$$

  where $\boldsymbol{m} = (m_1, m_2, \cdots, m_d)$ is a vector that uniquely identifies which joint moment we are dealing with.

- We can compute estimates of the underlying density directly at a point $\mathbf{p}$ in our measurement space by counting the number of data points that appear in a neighborhood around $\mathbf{p}$.

Using this information as our starting point, there are various approaches we could take. We can conceivably compute the density of each and every point in our domain, thereby producing a histogram with infinitesimally small bins. Doing this requires a clever, continuous version of a histogram. This is well known and studied in the literature as the kernel density estimator [34], and it does a reasonably good job at estimating the true density. It does, however, suffer from some problems, notably that it is extremely computationally intensive, and it tends to show evidence of lobes

and bumps that do not necessarily exist in the real pdf. Because the kernel density estimator is so well known and simple, we will use it as the basis of comparison for the technique we will propose.

Another approach is to compute as many moments and joint moments as possible from the data, and attempt to somehow convert the information into a probability density function. This second path is also very computationally intensive, because to completely specify a pdf, we need an infinite number of moments, even if we are only working with one dimensional data. Even if we only want to compute up to κ order moments, in d dimensions, there are $(\kappa+1)^d$ of them. For example, with fifth order moments, in four dimensions, we would need to compute more than one thousand joint moments and come up with a function that satisfies all of them.

If at first glance one is not deterred by the sheer volume of computations, this moment-method type approach is conceptually appealing. We could use a computer to estimate hundreds of moments, call them $\xi_{\boldsymbol{m}}$. Using the moment identifier $\boldsymbol{m} = (m_1, m_2, \cdots, m_d)$, we can define our notation such that $\mathbf{x}^{\boldsymbol{m}} = x_1^{m_1} x_2^{m_2} \cdots x_d^{m_d}$, $\mathbf{x} = (x_1, x_2, \cdots, x_d)$. Then, for each joint moment, we have a value of $\boldsymbol{m}$ and we can write the simple equation

$$\xi_{\boldsymbol{m}} = \int \mathbf{x}^{\boldsymbol{m}} f(\mathbf{x}) d\mathbf{x} \tag{5.2}$$

Now we have a tremendous set of $(\kappa+1)^d$ non-linear integral equations, one for each $\boldsymbol{m}$, that we would like to solve simultaneously. The only unknown in all of these equations is the probability density function $f(\mathbf{x})$.

Disappointingly, this set of integral equations turns out to be ill conditioned. This

is because for any finite set of moments, there are an infinite number of densities that satisfy the equations. We could approach this numerically, and perform singular value decomposition to find a set of basis vectors that spans our solution space, but then how do we choose from among them?

The situation at this point has many problems with it, but a concept called entropy will rescue the whole idea. Entropy is a measure on the space of probability density functions, which will help us choose a good density.

To understand why the measure of entropy is useful, assume we call the space of probability density functions S. For a finite set of constraints of the form

$$\xi_m = \int \mathbf{x}^m f(\mathbf{x}) d\mathbf{x} \tag{5.3}$$

there is a family of pdfs in S that satisfies those constraints – call this set S_c. Within S_c, there is a probability density function, $f_m(x)$, which maximizes the entropy. One of the properties of entropy is that most of the other points in S_c are clustered tightly around $f_m(x)$, and thus most of the points in set S_c are close to $f_m(x)$, by some measure of distance between two functions.

In the absence of any supportable criterion with which to choose the specific density in set S_c that most closely represents our data set, the maximum-entropy function provides the best choice we can make. It is, on average, closest to the actual, underlying density that created the data.

Theoretically, we can use the maximum-entropy idea to find a functional form, which both satisfies as many moments as we want and enforces the maximum entropy constraint to find a unique solution.

The next glaring problem that we need to deal with is the sheer number of moments we need to compute to characterize our data. As it turns out, the maximum entropy formalism is fairly general, and doesn't specifically need to be given the moments as constraints. If instead, we estimate the density for a grid of points (simply looking at how many data points are in the vicinity) and use these as constraints with the functional form we found above, we find that we produce excellent results.

5.1.2 Literature on Density Estimation

Density estimation techniques can broadly be divided into the parametric versus the nonparametric techniques. The parametric density estimation techniques assume some knowledge of the density underlying the data, such as: we know it is Gaussian, but we just do not know the mean or standard deviation. With this assumption, the object is then to find the best mean and standard deviation such that a Gaussian pdf would best fit the data. Nonparametric techniques, on the other hand, attempt to allow the data to specify what type of shape the pdf has.

A survey of the literature reveals many existing techniques. Some of the mainstream nonparametric techniques are summarized below.

- The histogram approach is the oldest and best known of all the techniques, and simply divides the data domain into bins, and measures the relative frequency of data inside each bin. This provides a rough estimate of the local density inside a bin. Histogram bins are typically the same size, but can be variable. Occasionally, this is called a variable partition estimate when the bins sizes are

varied.

- The kernel density estimator is a smooth function which is the sum of many Gaussian pdfs, one centered at each data point. This is actually a cleverly designed histogram whose bins are infinitesimally small, and where each point contributes a fractional count to many surrounding bins. This fractional count is larger for bins that are closer to the point, and smaller for those further away. There are many variations on the kernel density estimator which attempt to vary the kernels depending on the local density. The simple kernel density estimator is the technique that is used as the basis for comparison in this research. Kernel density estimators are introduced and discussed in many books such as [34] and [9].

- The orthogonal series estimator assumes that the histogram can be represented by a set of orthogonal basis functions. It attempts to estimate the coefficients of that decomposition by using the data. An introduction to the orthogonal series estimator is presented [34].

- The maximum penalized likelihood estimator attempts to maximize the likelihood of the density based on the data. Left alone, maximum likelihood techniques will yield a set of delta functions, so a smoothing requirement is added (thus the penalized part of the technique), which yields a density estimate. An introduction to the maximum penalized likelihood estimator is presented [34].

- The nearest neighbor estimator assigns densities based on bins centered on the

region of interest, which contain a fixed number of points. Thus, the bin sizes vary inversely with the value of the density. This technique is also described in [34].

Details of these techniques, as well as a wealth of other sources can be found in [34] and additionally [9]. These techniques typically have a sort of ad–hoc flavor, and most are not easy to generalize to higher dimensions in a simple way. Even recent papers such as [8] do not often treat multivariate densities. The book by Scott [32] has both discussion as well as other sources of information on multivariate density estimation.

Edwin T. Jaynes has written many well thought–out papers on the concept of maximum entropy, such as [14, 15]. Other articles on maximum entropy include [11] and [37].

There have been many efforts to use maximum entropy in the context of density estimation, that can be found in the literature [5, 12, 3, 4, 28]. Most closely related to our investingation is the research by Borwein *et. al* in [3], in which they successfully demonstrate maximum-entropy density estimation of different data sets. However, the technique by which they solve the entropy-moment problem is significantly different than the one we will describe.

We propose to offer a novel algorithm, one that is generally applicable and computationally efficient, which solves the maximum entropy density estimation problem for multivariate densities, and provides an analytical solution that is practically useful.

5.2 Theory

We would like to take a set of measured data, and come up with a best estimate of the probability density function that generated it. We will choose constraints, solve for all possible densities that satisfy them, and then choose the most likely density from the family of solutions by applying maximum entropy techniques.

Maximum entropy techniques are so called because they try to choose the density with the maximum entropy from a set of probability density functions. The concept of maximum entropy is not new, but the application to density estimation using this method, is.

5.2.1 What is entropy, and why maximize it?

Entropy is simply a scalar valued function of a probability density function. Thus, if $\mathbf{p} = (p_1, p_2, \cdots, p_n)$, is a discrete density function, and p_i is the probability that event i occurs, the entropy is usually written as

$$H(\mathbf{p}) = -\sum_{i=1}^{n} p_i \log p_i \tag{5.4}$$

The best way to describe the utility of entropy is to use Jaynes' own words [15]:

> "For many decades it has been recognized, or conjectured, that the notion of entropy defines a kind of measure on the space of probability distributions, such that those of high entropy are in some sense favored over others. The basis for this was stated first in a variety of intuitive forms: that distributions of higher entropy represent more "disorder," that they are

"smoother," "more probable," ... and that they "assume less" according to Shannon's interpretation of entropy as an information measure, etc."

We will apply maximum entropy techniques to choose among the possible densities that satisfy our constraints as described earlier. The densities so selected are smoother, more spread out, and the process does not assume more about the statistics that produced the data than it absolutely has to.

5.2.2 Why does maximizing entropy produce the most likely pdf?

In the following section, we take the idea that most of the solutions of interest are clustered around the maximum-entropy solution, and make it precise. This section summarizes mathematical arguments and intuitive arguments presented by Jaynes in [15].

Given a set of observations of a multivariate random variable, the following section proves that of all the densities that could give rise to those observations, a large fraction F of those densities have an entropy between $H_{\max} - \Delta H$ and $H_{\max}$, where ΔH is small.

Consider a random variable that has n possible outcomes. Now construct an experiment that consists of observing this random variable N times. We would like to know the probability density function $\mathbf{p} = (p_1, p_2, \cdots, p_n)$ that gave rise to the observations. Here, each p_i is the probability that the random variable produced outcome i. We can estimate p_i if N_i is the number of times outcome i was observed, and

simply write $p_i = \frac{N_i}{N}$. This is simply an estimate of the true density function, made by computing different frequencies of occurrence of each outcome in the experiment.

Conceptually, we can arrange all the possible probability density functions in an n-dimensional space, representing each probability density function as a point. Call this set S, such that for each n-dimensional point $\mathbf{p} = (p_1, \cdots, p_n)$ in S, each $p_i \geq 0$ and $\sum_{i=1}^{n} p_i = 1$.

For each point $\mathbf{p}$ in S, we can define a weighting $W(\mathbf{p})$, which represents the number of ways the experiment can give rise to that probability density.

$$W(\mathbf{p}) = \binom{N}{N_1}\binom{N - N_1}{N_2}\cdots\binom{N - \sum_{i=1}^{n-1} N_i}{N_n} \tag{5.5}$$

The first term in this equation is telling us that there are $\binom{N}{N_1}$ ways of obtaining the first outcome N_1 times. Having used up N_1 observations, there are now $N - N_1$ left. This means there are $\binom{N-N_1}{N_2}$ ways left to obtain the second outcome N_2 times. And so on.

We let $N \to \infty$ and use Sterlings approximation ($m! \approx (m/e)^m$) to write

$$W(\mathbf{p}) \longrightarrow \frac{\left(\frac{N}{e}\right)^N}{\prod_{i=1}^{n}\left(\frac{N_i}{e}\right)^{N_i}} \tag{5.6}$$

Taking the logarithm as $N \to \infty$,

$$\frac{1}{N}\log W(\mathbf{p}) \longrightarrow -\sum_{i=1}^{n} p_i \log p_i = H(\mathbf{p}) \tag{5.7}$$

Now each point $\mathbf{p}$ in S has an entropy $H(\mathbf{p})$ associated with it, where this 'entropy' is a monotonic function of its weighting factor $W(\mathbf{p})$.

The points in S are laid out in a regular n-dimensional cube. It can be shown that the entropy associated with each point would attain the maximum value of $\log n$ at

the center of the cube, and decrease monotonically as the distance from the maximum increases. The points (probability density functions) at the corners would have the minimum value of zero.

Assume we want to constrain our probability density function so that it satisfies the equation $\sum_{i=1}^{n} a_{ij} p_i = d_j$. Constraints of this form commonly come from specifying moments of the distribution. Note that we have already specified a constraint of this form $\sum_{i=1}^{n} p_i = 1$ to ensure that our **ps** are probability density functions. Each constraint specifies a subset of the original n-dimensional cube. The intersection of those subsets are the densities that satisfy the constraints we have.

Each constraint can be seen as reducing the dimensionality of the remaining possibilities by one. Thus, if we have m constraints (including the pdf condition) and the space of possibilities has n dimensions, we can rearrange the remaining possibilities in an $(n - m)$-dimensional space.

Because of the monotonicity that was noted above, after the constraints are enforced, there are two lessons we can deduce. First, there is a new unique maximum entropy point, which we will call $\mathbf{p}_m$ which has an entropy $H_{\max} = H(\mathbf{p}_m)$. Second, if we center our coordinate space around this new maximum density, the entropy is still monotonic and decreasing. We can take the first term of a Taylor series expansion around the new origin and write

$$H(\mathbf{p}) = H(\mathbf{p}_m) - ar^2 \tag{5.8}$$

where $r = |\mathbf{p} - \mathbf{p}_m|$ and a is a positive constant.

Let us now restate the problem and purpose of this proof: we have run an experi-

ment which consisted of observing a random variable N times. We want to know the probability density function that gives rise to the data we are observing. Our data and prior knowledge are provided as constraints. We also assume that what happens next must be consistent with what happened before.

Of all the probability densities which satisfy the constraints, we would like to find the fraction of those probability densities, F, whose entropy is within a small ΔH of the maximum entropy $H_{\max}$.

To do this, we define the following integral:

$$I(R) = \int_0^R \left[\frac{W(\mathbf{p})}{W(\mathbf{p}_m)}\right] r^{(n-m-1)} dr \tag{5.9}$$

Upon using Equation 5.7 and taking $N \to \infty$, we have

$$I(R) \longrightarrow \int_0^R \left[e^{N(H(\mathbf{p})-H(\mathbf{p}_m))}\right] r^{(n-m-1)} dr \tag{5.10}$$

$$= \int_0^R \left[e^{-Nar^2}\right] r^{(n-m-1)} dr \tag{5.11}$$

The term in brackets is a monotonic function of the number of ways to realize a particular distribution, which is distance r away from the maximum. The quantity on the right $r^{(n-m-1)}dr$ is proportional to the volume of an infinitesimally thin shell in $(n-m)$-dimensional space. This volume can be seen as a measure of how many points have the weighting given by the bracketed term.

The fraction we are looking for can now be written as $I(R)/I(\infty)$. Using a change of variables, $x = 2Nar^2$, we recognize this ratio as the cumulative distribution of a random variable with a χ^2 distribution.

Thus the fraction F of densities between $H_{\max}$ and $H_{\max} - \Delta H$ is given by

$$\chi^2_{(n-m-1)}(1-F) = 2N\Delta H \tag{5.12}$$

where 1-F is the area under the curve, as shown in the diagram explaining the notation.

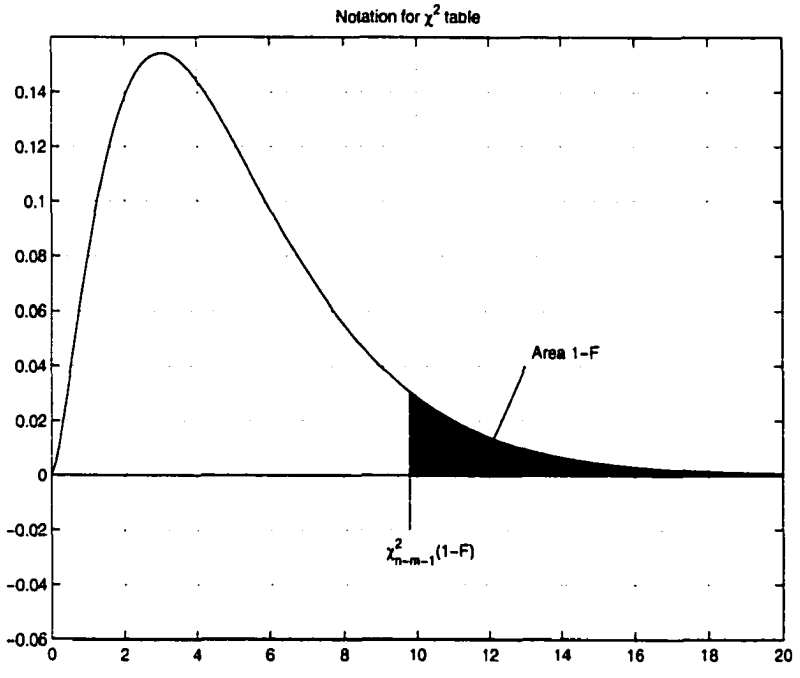


Figure 5.1: A χ^2 distribution with n-m-1 degrees of freedom with $\chi^2_{n-m-1}(1-F)$ labeled.

In [15], this relationship is called the Concentration Theorem. It is interesting to note that a large fraction of the possible distributions are within a very small ΔH of $H_{\max}$.

Thus, given the constraints we have, there is only an infinitesimally small chance that the distribution is near a **p** with low entropy. Thus, it is prudent to choose the maximum entropy distribution as our best guess.

5.3 Solution

This section demonstrates how we would apply maximum-entropy ideas to create a density estimation technique, and begins by setting up the well known entropy-moment problem and proposing a unique approach to solving it.

5.3.1 The Entropy-Moment Problem

We begin the density estimation procedure with a set of multivariate data taken from some random process. We would like to find the probability density function that most likely produced that data.

For conceptual clarity, our example is in one dimension, but we will find it simple to generalize to higher dimensions when the need arises.

We begin by estimating a finite set of moments, and writing them, along with the analytical expression that would produce each one:

$$\int f(x)dx = 1 \tag{5.13}$$

$$\int xf(x)dx = \xi_1 \tag{5.14}$$

$$\int x^2 f(x)dx = \xi_2 \tag{5.15}$$

$$\int x^3 f(x)dx = \xi_3 \tag{5.16}$$

$$\vdots$$

where the ξ_ns are the estimated moments, and the only unknown in this set of integral equations is $f(x)$.

As we mentioned before, a finite set of integral equations of this form do not have a unique solution. There is a family of solutions that satisfies them. We would like to simultaneously satisfy these equations and maximize the entropy. This will produce a unique solution to our problem.

Because it can be proven that $H(f(x))$ is a concave function[2] of $f(x)$, we can be confident that there exists an f_m that maximizes the value of $H(f(x))$ and satisfies our requirements.

To maximize $H(f(x))$ subject to our constraints, we can use the method of Lagrange multipliers to maximize

$$H(f(x)) + \int \lambda_0 f(x)dx + \int \lambda_1 x f(x)dx + \int \lambda_2 x^2 f(x)dx + \ldots \tag{5.17}$$

which can be written as

$$\int \overbrace{\left[-f(x)\log f(x) + \sum_n \lambda_n x^n f(x)\right]}^{Q(x,f(x))} dx \tag{5.18}$$

Define the integrand as $Q(x, f(x))$, and define $f(x) = f_m(x) + \epsilon\eta(x)$, where $\eta(x)$ is any well behaved function we choose, and ϵ is a constant.

[2]Here, we present the proofs using a continuous version of entropy, which rather than being a function of a discrete probability density function p_i, is a function of a continuous probability density function $f(x)$.

The continuous version of entropy can be written as

$$H(f(x)) = -\int_{-\infty}^{\infty} f(x)\log f(x)dx$$

The proofs of the last section (Concentration Theorem) follow analogously for the continuous version of the entropy, and the proofs of this section can be rewritten for the discrete version of entropy.

When $\epsilon = 0$, $f(x)$ maximizes the integral of interest, which can be written as:

$$\frac{d}{d\epsilon}\int Q(x, f(x))dx\Big|_{\epsilon=0} = 0 \tag{5.19}$$

$$\text{by Liebnitz Rule, } \int \frac{\partial}{\partial\epsilon}Q(x, f(x))dx\Big|_{\epsilon=0} = 0 \tag{5.20}$$

$$\int \left[\frac{\partial}{\partial f(x)}Q(x, f(x))\right]\frac{\partial f(x)}{\partial\epsilon}dx\Big|_{\epsilon=0} = 0 \tag{5.21}$$

$$\int \left[\frac{\partial}{\partial f(x)}Q(x, f(x))\right]\eta(x)dx\Big|_{\epsilon=0} = 0 \tag{5.22}$$

Setting $\epsilon = 0$ is the last necessary step, and we note that it does not change the integral at all. The conclusion we draw is that since $\eta(x)$ is an arbitrary, well behaved function, the maximum $f(x)$ must satisfy $\frac{\partial}{\partial f(x)}Q(x, f(x)) = 0$. Solving this, we find

$$f(x) = e^{P(x)} \tag{5.23}$$

where $P(x) = \sum_{n=0}^{\infty}\lambda_n x^n$. Note that this polynomial could be a Taylor series expansion of an arbitrary function, and that each term in the polynomial corresponds to a single constraint. Thus, there are as many unknown coefficients (λ_n) as there are integral constraints.

We substitute $f(x) = e^{P(x)}$ into the original set of equations,

$$\int e^{P(x)}dx = 1 \tag{5.24}$$

$$\int xe^{P(x)}dx = \xi_1 \tag{5.25}$$

$$\int x^2e^{P(x)}dx = \xi_2 \tag{5.26}$$

$$\int x^3e^{P(x)}dx = \xi_3 \tag{5.27}$$

$$\vdots$$

This is the entropy-moment problem, which generalizes easily to higher dimensions.

5.3.2 Solving the Entropy-Moment Problem

Despite this natural, conceptually simple treatment of density estimation using maximum-entropy ideas, solving for the coefficients of $f(x)$ is not computationally trivial. A direct solution to these equations can be found in Borwein and Huang in [3].

We will now describe a novel solution to the entropy-moment problem which solves for the unknown coefficients in a computationally efficient, accurate manner.

The solution we propose can be conceptually explained by making an analogy with a multi-dimensional Fourier transform representation. In the terminology of the Fourier transform, we first sample the function in its domain, and then transform it to its frequency domain representation.

The samples of the function in its domain for our analogy correspond to a histogram representation we will produce from the data set. The basis functions we use, rather than being complex exponentials, are Legendre basis functions. The frequency domain representation in our analogy still corresponds to a weighting of our basis functions, but these weighting coefficients to the Legendre basis functions are exactly the coefficients that we seek in our solution.

This transformation, that we will call the Legendre transform, will convert a histogram representation of the data into the coefficients that we seek.

5.4 Implementation

We will now describe the implementation details of the maximum-entropy density estimation technique that we developed, from the histogram representation to the details of the Legendre transform discussed in the previous section.

We will enforce the requirement of maximizing entropy by fixing the functional form of our pdf, as given by Equation 5.23, and also ensure that the probability density function, if sampled, closely matches the probability values on the histogram grid.

5.4.1 Developing the Histogram

The first step in implementing the density-estimation technique we outlined above is to constrain our solution with the integrals in Equation 5.16 with a histogram of the sampled data. From this histogram representation, we get a noisy estimate of density on a grid of points.

Histograms are produced by dividing the domain of the function into regularly spaced bins, and each bin is assigned a value based on how many points it contains. The result is a piecewise continuous curve with a derivative of zero everywhere. The bin spacing is chosen so as to adequately sample the domain of our measurements.

Typically, a histogram is presented for functions of one variable, but in our case the histogram is multidimensional. This does not change its basic construction or characteristics. The domain is still divided up into regularly spaced bins, but now the bins in the domain are no longer segments of the number line. Instead they are

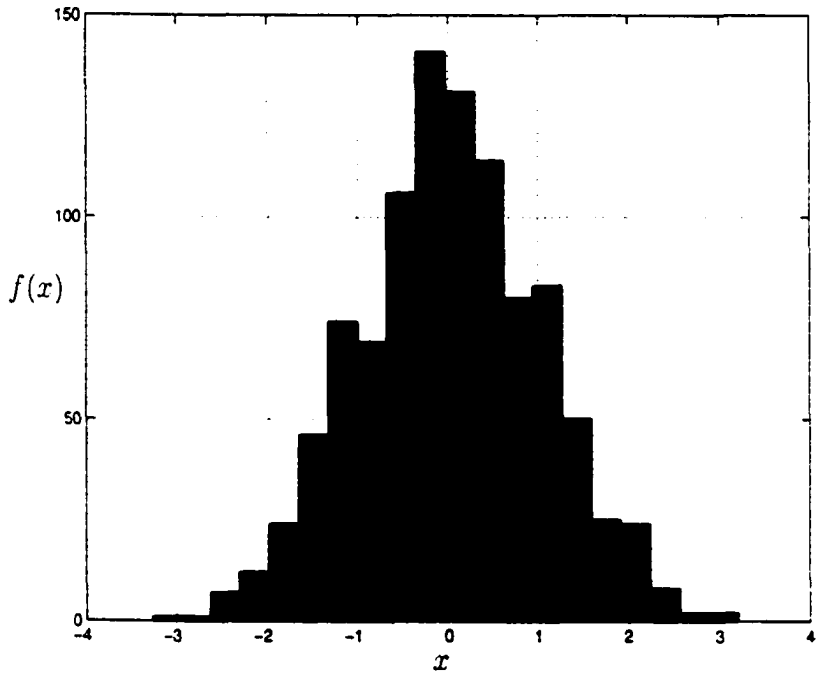


Figure 5.2: A one-dimensional histogram.

multidimensional cubes which tile our domain.

At this point, the histogram is best described by a series of points, each describing the center of a cube. With each point, we pair a scalar value, which is our best estimate (based on the data we have) of the density function inside that cube.

There is tremendous computational inefficiency that crops up here, usually called the curse of dimensionality. Looking through ten bins to find how many data points are in each bin might be feasible in a one dimensional domain, but if you have a multidimensional domain the computational expenses increase exponentially. For a four dimensional domain, dividing each dimension into ten bins, there are 10^4 bins to search through.

Luckily, in most cases, there are large regions in our domain where there is a nearly zero probability that we measure any data there. We can exploit these regions

by reversing the focus of our search. Instead of looking for which data points are in a particular bin, we can look at which bin each data point is in. This increases search efficiency by orders of magnitude.

To do this, we take each point and compute the center of the bin that contains it. If $\mathbf{a} = (a_1, a_2, \cdots, a_m)$ is our data point, l_d is the minimum value of our domain in dimension d, and s_d is the length of a side of the cube in dimension d, then

$$c_d = \left\lfloor \frac{a_d - l_d}{s_d} \right\rfloor s_d + \frac{s_d}{2} \tag{5.28}$$

is the d^{th} coordinate of the center of the cube that the point falls in. In this way we only need to instantiate and update data structures for the cubes that contain data. In practical terms, it is sometimes useful to remove outliers from the data set. Outliers are simply data points in a region of the density that is sparsely characterized and far away from the center of the domain.[3]

Each cube's scalar value is determined by counting the number of points inside it (n), dividing by the total number of points in our data set (N), and normalizing it so that the whole histogram integrates to one. The value of each cube is given by $(n/N)(\prod_d s_d)^{-1}$.

5.4.2 Implementing the Legendre Transform

We can use the histogram as a rough density estimate, but it has many jagged edges and this introduces an error into our representation that may be unacceptable.

Instead of looking at the histogram as a set of cubes, consider our histogram as

[3]If you find yourself removing more than a couple outliers for a particular data set, then they are not really outliers.

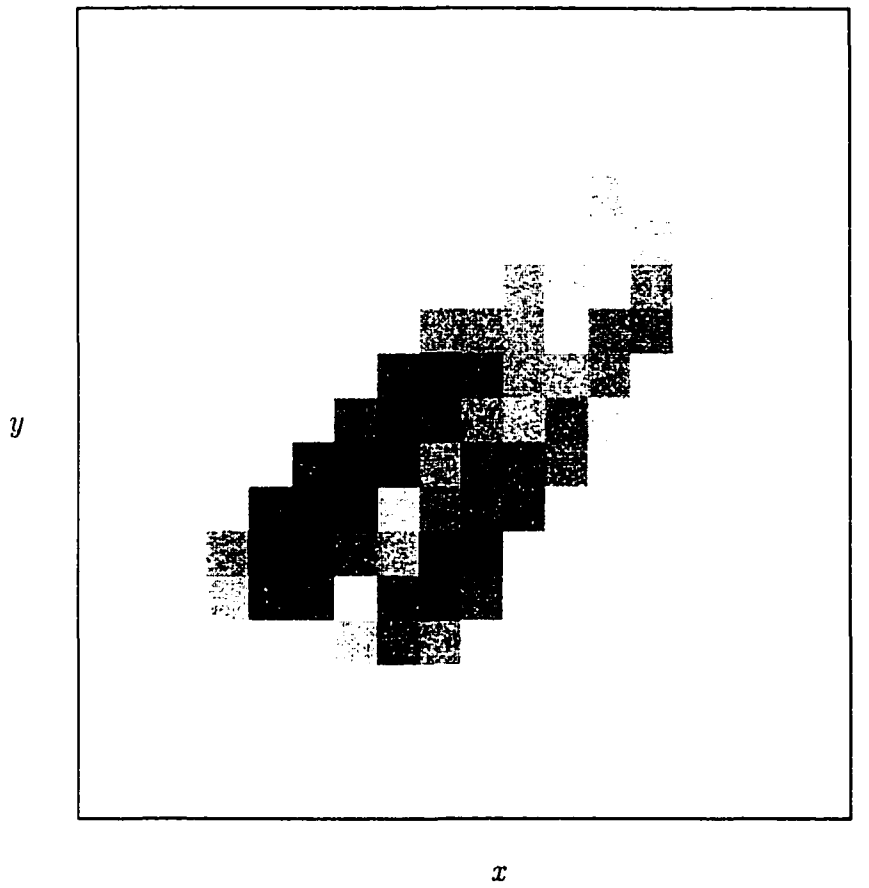


Figure 5.3: A two-dimensional histogram, viewed from above. The height of each bin is indicated by shading.

representing a sampling of our density function at particular points (the cube centers). We now have a regular grid of points on which have estimated our probability density function.

We would like to find the set of coefficients that produce a probability density function that maximizes the entropy, while still taking on the values at the points we have sampled. Before, we described this solution as a transformation, since we are converting samples to weighting coefficients for orthogonal basis functions.

The approach we are taking is to examine one dimensional slices from the sampled density and expand them in terms of convenient basis functions. The coefficients

of these basis functions can easily be collected and expanded in terms of the next dimension.

Thus, if we were working with a two dimensional probability density function, whose orthogonal axes were x and y, the first expansion produces many densities of the form $e^{P_0+P_1x+\cdots+P_\kappa x^\kappa}$, one for each set of (y, z) coordinates. We can collect these together, and each coefficient can again be expanded in terms of basis functions, so we could write $P_0 = \sum_n R_{n,0}\Phi_n(y)$, $P_1 = \sum_n R_{n,1}\Phi_n(y)$, etc .... For higher dimensional densities, the process can be repeated, expanding the existing coefficients in terms of the next dimension's variable.

Let us be more precise about this algorithm, which does most of the work of converting a data set into the analytical density that we desire.

Definitions

First, we present some definitions that will allow a clearer explanation of the algorithm that follows. If a vector is a list of numbers, we define the *tail* of the vector as the last entry in the list. We define the *body* as the vector without its tail. Thus, given a vector $v = (v_1, v_2, \cdots, v_{n-1}, v_n)$ the body is given by $v_{\text{body}} = (v_1, v_2, \cdots, v_{n-1})$ and the tail is given by $v_{\text{tail}} = (v_n)$.

Our n-dimensional histogram is made up of units that consist of a vector describing the location of a point in the pdf's n-dimensional domain, and a scalar describing the estimate of the density at that point.

We will call these units stars, because we can initially imagine them as points with a particular brightness, like the stars in the sky. We will call the vector within each

star the "location vector" while we will call the scalar the star's "value multigrid." This terminology seems cumbersome at first glance, but it allows us to more easily describe the algorithm while being consistent with the names of classes and objects in the C++ code that implements the algorithm.

The value for each star is actually what we call a multigrid - a multidimensional grid of numbers, something like a higher order tensor. For the histogram or sampling of our density, each bin is a star, and the value of each multigrid is simply a scalar. The location vectors select points in the domain of our density, so for a d–dimensional density, the location vectors are d–dimensional, also.

The Reduction Algorithm

The reduction algorithm is the algorithm by which the histogram is reduced to a grid of coefficients that, when used with $f(\mathbf{x}) = e^{P(\mathbf{x})}$, describe a smooth, analytical maximum entropy density function.

We will describe the reduction algorithm, and along with our description, give a two dimensional example. For this example, we used the data set pictured in Figure 5.4.

Our first step is to convert the data set into a histogram representation. Graphically, we can imagine the histogram looking like Figure 5.5, and as a sparse numerical representation, we can visualise it as in Table 5.1.

At this point, each histogram bin corresponds to a star. The reductions algorithm is described below.

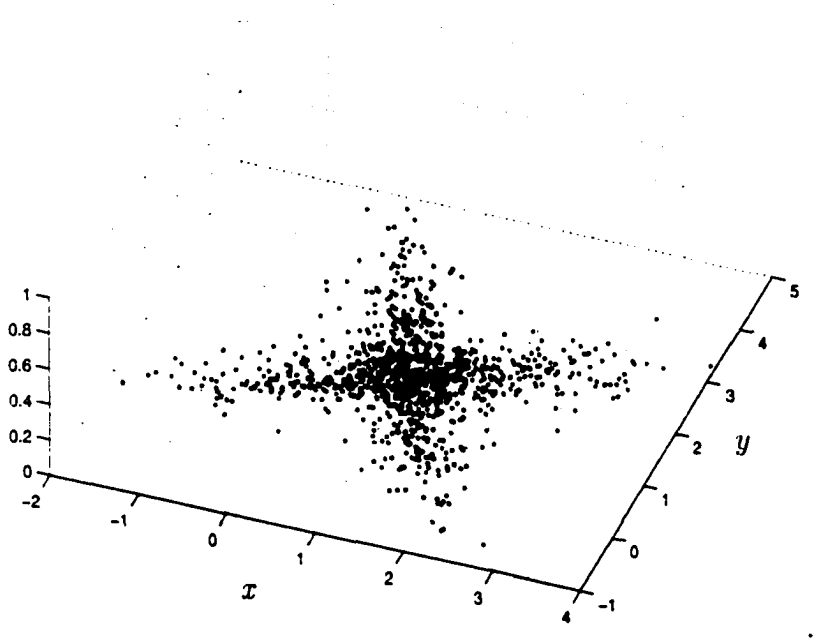


Figure 5.4: A data set of realization from a two-dimensional density.

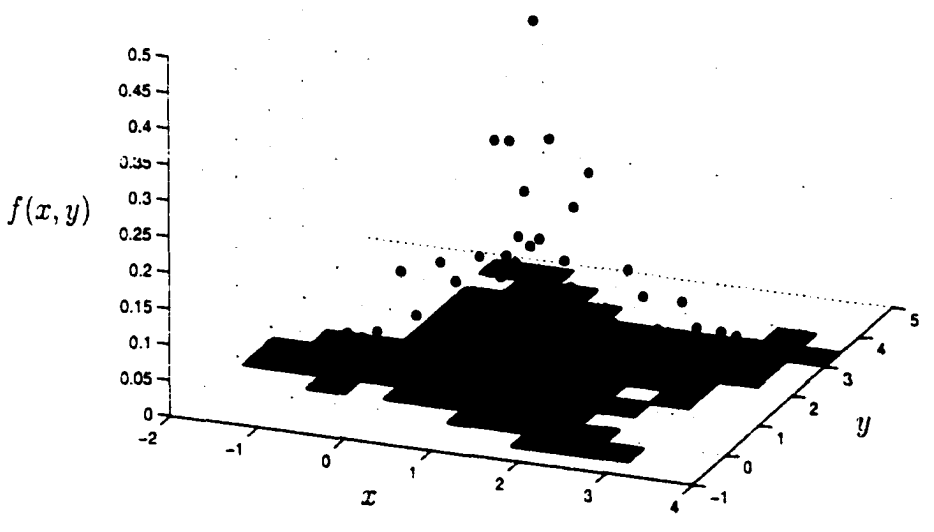


Figure 5.5: A graphical histogram representation of our data set. The height of each point above the domain corresponds to the level of probability that a data point is in the specified bin.

			0.009	0.004							
				0.009	0.004						
			0.128	0.148	0.004	0.004					0.014	
			0.049	0.104	0.163	0.019	0.014	0.004	0.009	0.004	0.014	0.004
			0.009	0.044	0.307	0.084	0.109	0.039	0.004	0.004	0.019	
			0.009	0.034	0.287	0.198	0.128	0.049	0.004	0.004	0.004	
		0.019	0.004	0.014	0.054	0.198	0.480	0.109	0.024	0.009	0.009	
	0.024	0.019	0.054	0.004	0.054	0.114	0.297	0.228	0.019		0.039	
	0.004	0.004	0.029	0.019	0.059	0.034	0.109	0.257	0.074	0.049		
			0.019		0.004	0.009	0.019	0.049	0.128			
							0.004	0.004	0.074	0.074		
									0.009	0.019	0.014	

Table 5.1: A numerical histogram representation of our data set.

1. Collect stars into a group, G, such that all stars whose location vectors have the same body b are in G.

   A single group G of stars is highlighted in Figure 5.6.

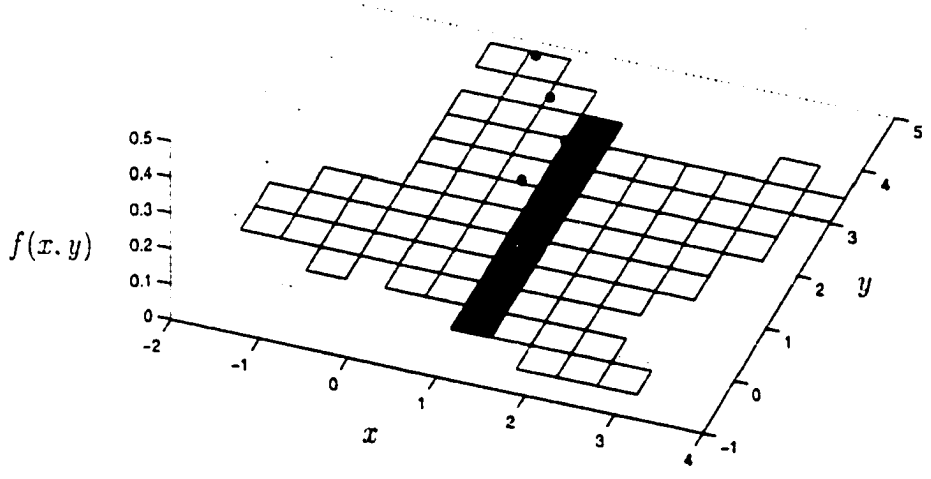


Figure 5.6: A one-dimensional slice of our histogram.

2. Create two vectors, one which is a concatenation of tails from the position vectors in this group (x), and the other is a vector of value multigrids (y_g). Plot these vectors in a cartesian coordinate system.

This corresponds to plotting the group G as shown in the top plot of Figure 5.7.

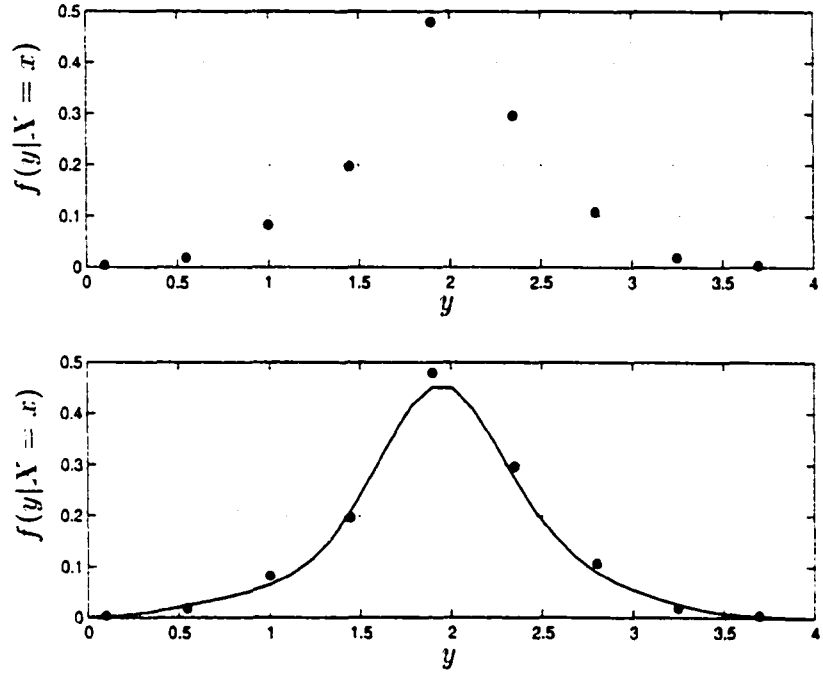


Figure 5.7: The top axis plots samples of the one-dimensional slice in Figure 5.6. In the bottom axis, we find the best set of weighting coefficients for our basis functions so that samples the final representation match the samples taken from the histogram. This can be thought of as a projection of a function defined by the original samples onto a Legendre basis representation.

3. Considering the one dimensional samples we have, we must answer the question of what weighting we must give to our Legendre basis functions so that a sampling would produce the results we see.

   Simple piecewise or even linear interpolation of these samples can produce a good estimate of the one dimensional function that they came from. This one dimensional function can then be decomposed into a set of coefficients that represent weightings of a set of basis functions chosen to span our solution space.

The bottom of figure 5.7 corresponds to the projection of the interpolated samples into the coefficient domain.

These coefficients are entered into a value multigrid which is one dimension higher than those in the group G. Graphically, we can think of our situation as in Figure 5.8

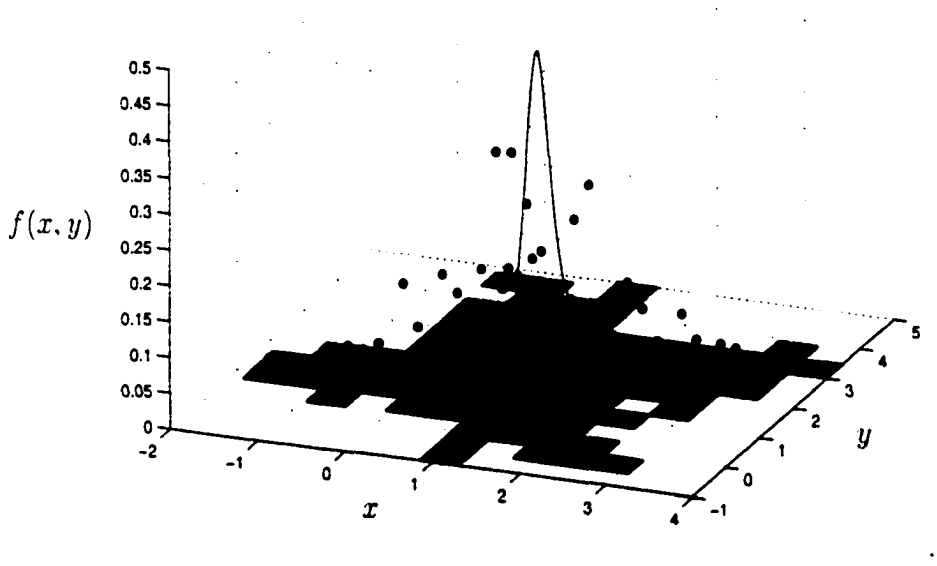


Figure 5.8: Our histogram, with a single slice of histogram bins converted into a continuous, one-dimensional function.

4. Go to step 1 and repeat for the next group G. After all the groups in our example are processed in this way, we can graphically visualize our situation as in Figure 5.9.

   Each one dimensional curve in Figure 5.9 is a star. Now, rather than a single coefficient, each star has a vector of coefficients, pictured in Table 5.2 as column vectors.

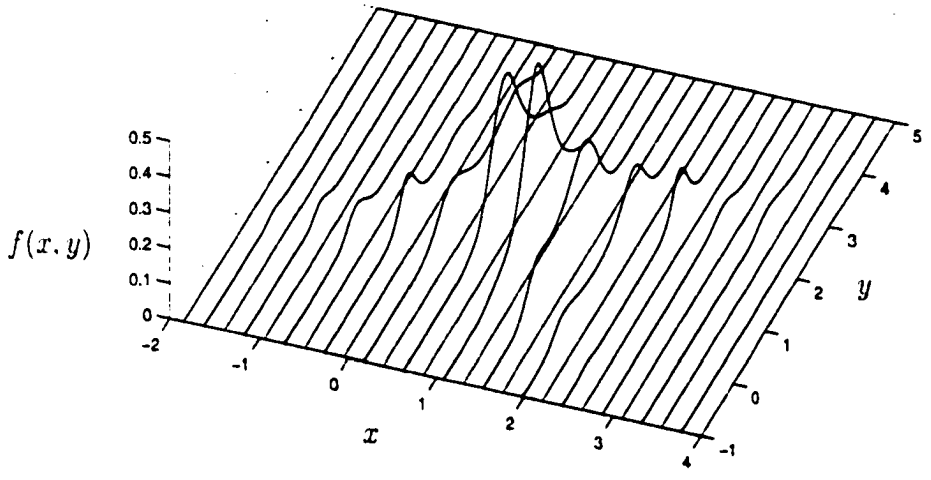


Figure 5.9: All slices of histogram bins have been converted into a continuous, one-dimensional functions.

0.0	0.0	0.7	0.4	-0.4	-1.0	-0.1	0.1	-1.1	0.3	0.0	-1.3	1.3	-1.5	-1.3	0.0	0.0
0.0	0.0	1.1	0.0	-0.5	-1.7	-0.4	-0.0	-0.4	-0.6	-0.2	1.1	-0.3	0.6	-1.3	0.0	0.0
0.0	0.0	-1.7	-1.1	0.1	-1.4	0.1	0.6	0.5	0.0	-0.5	-1.5	-1.9	0.3	-0.5	0.0	0.0
0.0	0.0	-0.3	0.9	0.3	2.1	-0.1	-0.3	0.9	-0.2	-0.2	-0.6	0.3	-0.3	1.0	0.0	0.0
0.0	0.0	2.2	1.3	0.6	3.0	-0.0	-0.9	-0.4	-0.5	0.2	1.2	2.1	0.3	1.2	0.0	0.0
0.0	0.0	-0.9	-1.9	-1.0	-0.4	-0.1	-0.4	0.0	1.3	1.2	0.6	-1.0	0.8	-0.2	0.0	0.0
0.0	0.0	-1.9	-0.3	-1.1	-0.6	-0.0	-0.1	-0.3	0.1	0.7	1.2	0.8	-0.9	-1.4	0.0	0.0
0.0	0.0	2.1	2.9	2.4	0.9	1.7	1.5	-1.8	-2.0	-2.6	-3.1	-0.9	-2.0	-4.7	0.0	2.0
0.0	0.0	0.3	-0.3	0.6	-2.1	0.4	0.6	0.5	0.1	-1.0	-2.3	-3.3	0.5	1.0	0.0	0.0
0.0	0.0	-2.5	-3.3	-3.5	-2.9	-2.5	-0.1	0.3	-0.1	2.5	5.2	4.3	3.4	1.6	0.0	0.0
0.0	0.0	0.9	2.6	1.2	1.3	-0.5	1.3	1.6	1.3	0.3	0.2	1.4	1.1	0.0	0.0	0.0
0.0	0.0	3.0	3.1	4.5	-0.6	-1.4	-1.9	1.3	3.3	2.1	-0.5	-1.4	-4.2	-2.3	0.0	3.0
0.0	0.0	-4.6	-5.7	-6.2	-7.3	-5.2	-10.0	-10.1	-9.7	-8.0	-6.1	-5.9	-5.9	-3.5	0.0	0.0
0.0	0.0	-1.5	-1.4	-2.3	2.3	2.5	1.4	-1.2	-2.4	-3.3	-2.1	-1.3	2.2	1.3	0.0	0.0
-15.0	-15.0	-11.2	-10.7	-10.0	-7.6	-6.2	-4.6	-4.6	-7.1	-6.9	-3.1	-4.6	-10.1	-11.3	-15.0	-15.0

Table 5.2: Each column is a vector of coefficients weighting Legendre basis functions for a single slice of the density. The blue hilighting is a slice of the coefficients used in the second pass of the reduction algorithm.

There are far fewer stars than before, and we have reduced the dimensionality of the location vectors by one, and increased the dimensionality of the value multigrid by one.

This reduction algorithm described here is repeated once for each dimension of the density we are estimating. The second time through the reduction algorithm, our slices are no longer in the density domain, but in the coefficient domain. A slice the second time the reduction algorithm is run looks like the highlighted numbers in blue in Table 5.2. Graphically, we plot them in Figure 5.10 as before and find weighting to our Legendre basis functions which describe the curve of interest.

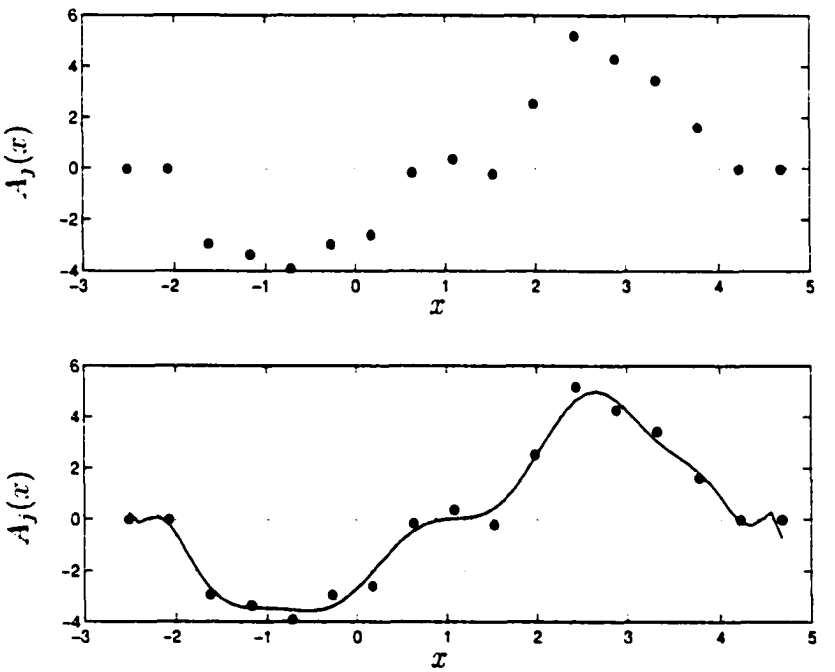


Figure 5.10: In the second pass of the reduction algorithm, slices are no longer in the density domain, the are in the coefficient domain. These curves correspond to the blue highlighting in Figure 5.2.

The final product of this procedure is a single star with a zero dimensional location vector, and a fairly complicated multigrid value. It defines the coefficients used in

our analytical function which specify the shape of the estimated density function. We must now normalize this density to ensure that it is indeed a probability density function.

In our two dimensional example, the reduction algorithm gets run only twice, converting our histogram into the grid of numbers in Table 5.11. These numbers are coefficients that describe our probability density function, pictured in Figure 5.12 as a two-dimensional surface.

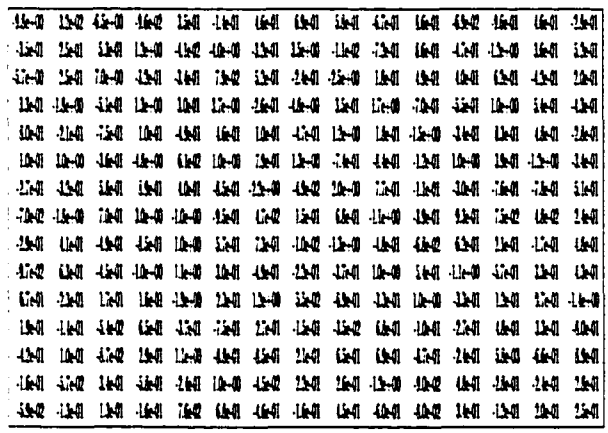


Figure 5.11: Final result of the reduction algorithm: a grid of coefficients which when used with our functional form, yield the surface in Figure 5.12.

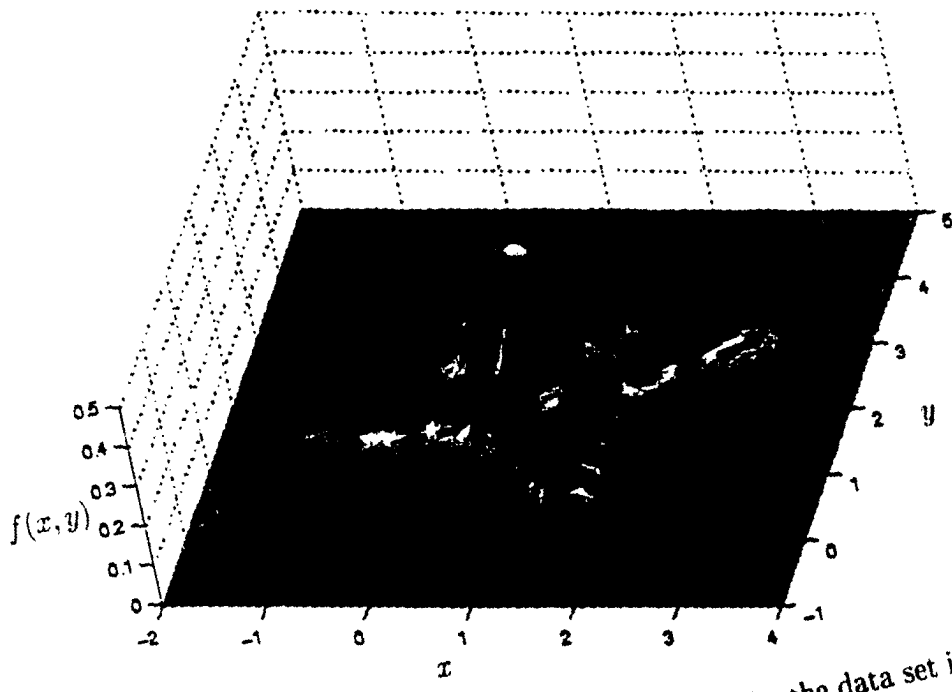


Figure 5.12: The estimate of the underlying density for the data set in Figure 5.4.

Choosing a set of basis functions

Some elaboration might be useful for step 3 in the reduction algorithm, where we talked about expanding a set of samples in terms of a set of basis functions.

For any set of samples of a one dimensional function $\{y_n\}$ which are taken at values $\{x_n\}$, we can use interpolative techniques to come up with an approximation which we will call $y_a(x)$. This approximation can be constructed using piecewise linear interpolation, or a spline interpolation, or any other more sophisticated technique available to us.

If we can choose an appropriate set of basis functions we can reasonably expect to be able to represent $y_a(x)$ in terms of weighted sums of them.

If the set of our basis functions is $\{\Phi_m(x)\}$, and they are orthogonal, then we can write

$$\int \Phi_n(x)\Phi_m(x)dx = \begin{cases} h_m^2 & m = n \\ 0 & otherwise \end{cases} \tag{5.29}$$

Orthogonality allows us to decompose $y_a(x)$ into a set of coefficients by simply integrating

$$a_n = \frac{1}{h_m^2}\int y_a(x)\Phi_n(x)dx \tag{5.30}$$

If we have appropriately chosen our basis, its projection will not be far off from the $y_a(x)$, and we can reconstruct our function from its coefficients by writing

$$\tilde{y}(x) = \sum_i a_i \Phi_i(x) \tag{5.31}$$

Thus, orthogonal or biorthogonal bases seem most promising, since we can easily expand an arbitrary function. This decomposition is actually a transformation into the coefficient domain of our basis functions. In the cases of interest to us, the Legendre polynomials are the theoretically optimal bases, since our functional form has a polynomial in the exponent. Thus, we used the Legendre basis exclusively in this investigation.

The Pinning Problem

In the process of implementing the algorithm above, there are various details that we should make sure we take into consideration.

In step 1 of the reduction algorithm, we take a subset of stars that comprise our histogram, and we called it G. It should be noted that the group is a one–dimensional slice of our histogram. It contains all the stars along a line. The location vectors of all the stars have the same coordinates except for the last dimension, which varies. Bins whose values are zero are not represented for space efficiency purposes.

The reduction algorithm discussed above decomposes a sampled function into a series of coefficients, which correspond to a particular set of basis functions. In constructing the continuous function, it is important to specify which parts of the function are zero, since we are not explicitly doing this.

We would like to ensure that the function goes to zero everywhere we haven't

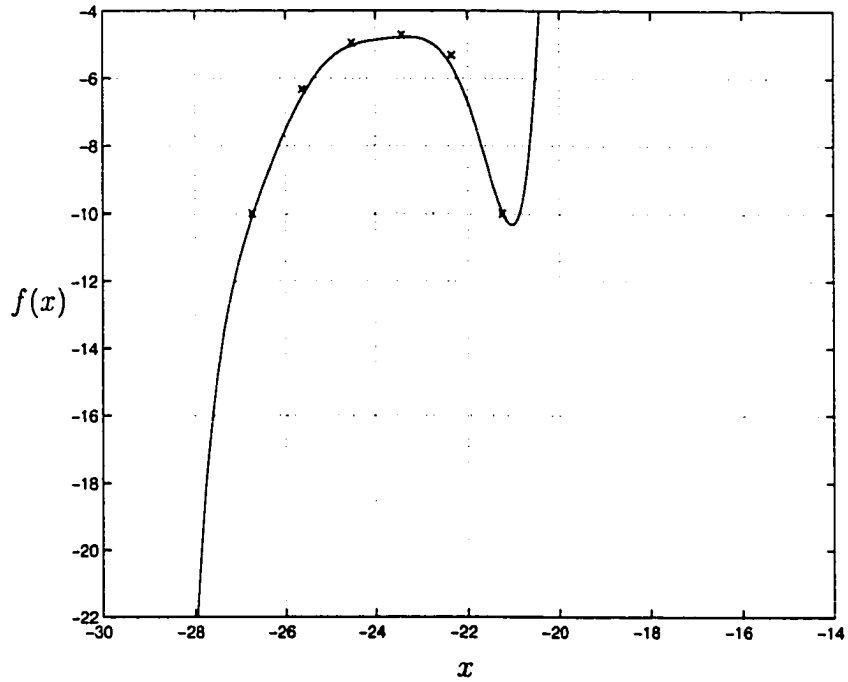


Figure 5.13: A one-dimensional slice of our probability density funcion. The x's in this graph are the measured samples (or histogram bin centers along with their values.) The approximated function described by our samples is decomposed into its basis coefficients, and then reconstructed and plotted over the entire domain. Note that this slice has been partially pinned, since the outer two x's are forcing the function to -10. However, the edges of the domain have not been pinned, and thus the odd polynomial does whatever it wants outside of the constraints. Thus, the need for pinning the function before decomposing it into basis coefficients.

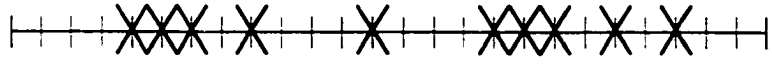
specified it, and in one or two dimensions, this is not a problem at all, since we can easily outline boundaries. However, because we are not assuming that the equiprobable surfaces of our density are simply connected, there may be disjoint blobs that we need to outline. Furthermore, who can tell the edges of a shape in more than three dimensions?

We need to pin down the edges of our density, and set them to zero at the boundaries. We are calling this the pinning problem, and here we outline a solution which generalizes nicely to higher dimensions.

The key to this solution is to use the idea of stars with higher dimensional value

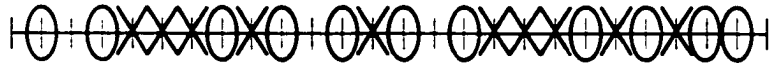
multigrids to pin the edges of a slice. Since a slice is always dealt with in one dimension, we can represent it as a number line, with the line representing the coordinate that is varying in the location vectors. Note that the stars are arrayed on a grid, but are not necessarily on every grid point.

The x's are star centers that are specified, and the o's are zero values. Things seem easy enough so far. If we begin with a number line like this:



we can "pin" the slice by adding zeros like this:

We also might need a separate stage which pins the edges of our domain in this dimension:



If we did not do this, then the transformation would find a function which sort of interpolates the gaps, and doesn't go to zero.

The problem occurs because pinning the overall function to zero is not the same as pinning the edge coefficients to zero. Before we ran the reduction algorithm, the coefficients are the density estimates in log space, so zero is a very large negative number.

After performing the reduction algorithm several times, not only are the value multigrids multidimensional cubes, but they are no longer in the domain of the density

estimates. A value of zero for a star with higher dimensional value multigrid depends completely on the basis functions we are using.

The solution to this is to start with a slice of our density that has a zero value and run the reduction algorithm on it. By doing this repeatedly, we can build up a multigrid, that when collapsed (see next section) evaluates the function at zero. We call these "zero multigrids." Building up these "zero multigrids" allows us to pin our function in coefficient space each time we run the reduction algorithm.

Collapsing the Multigrid

Once we have constructed a probability density function, we must be able to evaluate our function to find the density at a particular point, $\mathbf{x}$. The multigrid contains numbers that are related[4] to the coefficients of the polynomial $P(\mathbf{x})$ and the density function we are evaluating is $f(\mathbf{x}) = e^{P(\mathbf{x})}$. All we need to do is evaluate the functions of interest.

An orderly way to approach this is by using the following algorithm, which we will call collapsing the multigrid. We call it this because at each stage in the algorithm, we are evaluating our polynomail at a point, collapsing a series of coefficients that go with that dimension into a single number. This gives us a visual picture of what is going on.

To collapse the multigrid:

1. Since our multigrid is indexed by a vector of numbers (one for each dimension)

[4]The numbers in the multigrid must be multiplied with the appropriate Legendre polynomial to get $P(\mathbf{x})$.

let us call this vector i for index vector. Choose an index vector, fix the body of the vector, and allow the last index of i (i_{tail}) to vary, so the tail takes on all the possible values it can. Gather the coefficients in a group, and note that this grouping is akin to the grouping done in the reduction algorithm, in that it represents a slice of a multidimensional grid.

2. With this slice of coefficients, reconstruct the function at the specified point by taking a weighted sum of the basis functions we used in the decomposition. In our case these are Legendre polynomials. Enter this value into a multigrid of one lower dimension, and index it with i_{body}.

3. Go back to step one, repeating until we have exhausted all the possible i's.

4. Once we have exhausted all the indices, we have reduced the dimensionality of our multigrid by one. Repeat the whole procedure until the dimensionality of the entire multigrid is zero.

At the end of the evaluation algorithm, we have, by successive reconstruction of slices using our basis functions and coefficients, evaluated our function at the specified point.

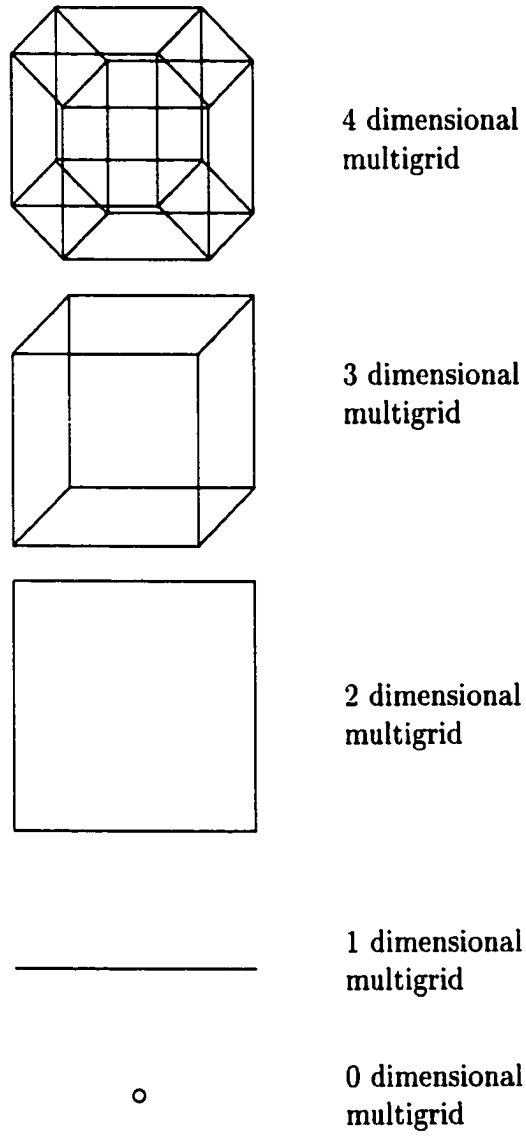


Figure 5.14: A visual depiction of the algorithm for collapsing the multigrid. This procedure is used when we are evaluating the density at a particular point. The progression is from top to bottom, and each figure is the geometric representation of a multidimensional grid of coefficients. At each stage, the dimensionality becomes smaller, because we are using the coefficients to evaluate the outermost dimension at the specified value. When the multigrid becomes zero-dimensional, the value that is left is the density at the point we have specified.

5.5 Results

In order to evaluate the performance of the maximum entropy density estimation technique, we took various oddly shaped densities that we knew very well, produced data sets from them, and then estimated their densities.

To see how close to the original density and final result were, the estimates were compared to the original densities using the following metrics: mean squared error, absolute error fraction, and the Kullback-Liebler distance measure. The simple kernel density estimate was also run on each of the data sets, and all of the same metrics were produced and compared to the maximum entropy techniques.

A theoretical comparison of the storage efficiency and the computational efficiency of the two density estimation techniques is also provided.

The results here are presented first visually, and then numerically (in table format). There are three test cases that were constructed, the X-density, the Y-density and the XV-density. The X and Y densities are two dimensional densities that are the mixtures of bivariate gaussians. The XV-density is a three dimensional density that can be thought of as a cylinder whose center is curved, giving it an overall banana-like shape. The cross section of this cylinder is always a mixture of the X and V densities, such that the cross section is the X-density on one end, and the V-density on the other end, and half of each in the middle.

5.5.1 Qualitative Comparison

Figures 5.15-5.26 present visual results of density estimates which constitute our qualitative comparison. The esimates are presented mostly for two dimensional densities, mainly because of the difficulty in visualizing three dimensional densities. Numerical results are presented in the next subsection for all densities that were tested.

5.5.2 Quantitative Comparison

This section quantifies the performance of the maximum entropy technique as compared to the kernel density technique. We consider various measures of the accuracy of the estimate, as well as computational efficiency and sparseness of representation.

Estimate Accuracy

The first concern we have is in measuring how close we have come to the original density we are trying to estimate. There are many ways to measure how different one function is from another. Our most important metric, and the one which the maximum entropy does well in, is the Kullback–Liebler distance. Not only is this distance metric specifically designed to measure the distance between two probability density functions, it is directly related (through Stein's Lemma, discussed in [7]) to the probability of error in certain classification problems.

The Kullback–Liebler distance is also defined in [7] as

$$D(f(x) \parallel g(x)) = \int f(x) \log \frac{f(x)}{g(x)} dx \tag{5.32}$$

Note that there are various difficulties with using the the Kullback–Liebler dis-

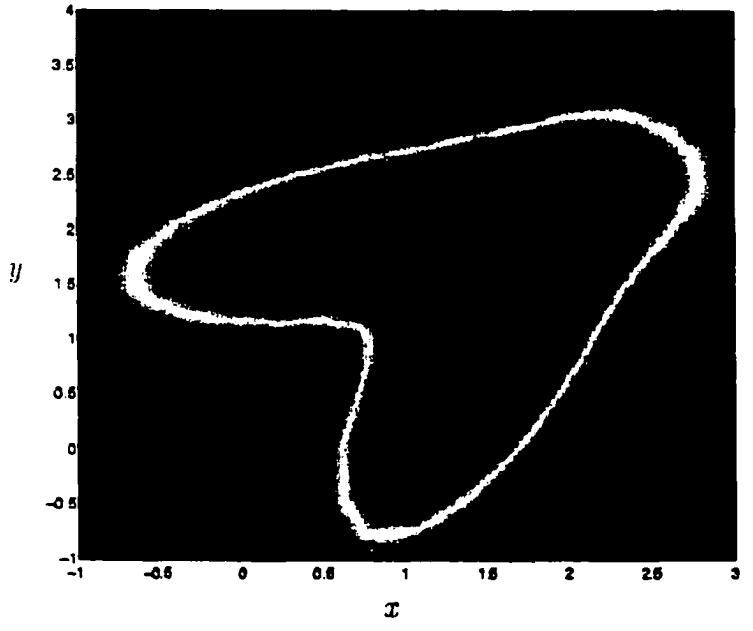


Figure 5.15: The theoretical density that we are taking our samples from. This is what we are trying to estimate. Because of its characteristic shape, we call it the V-density

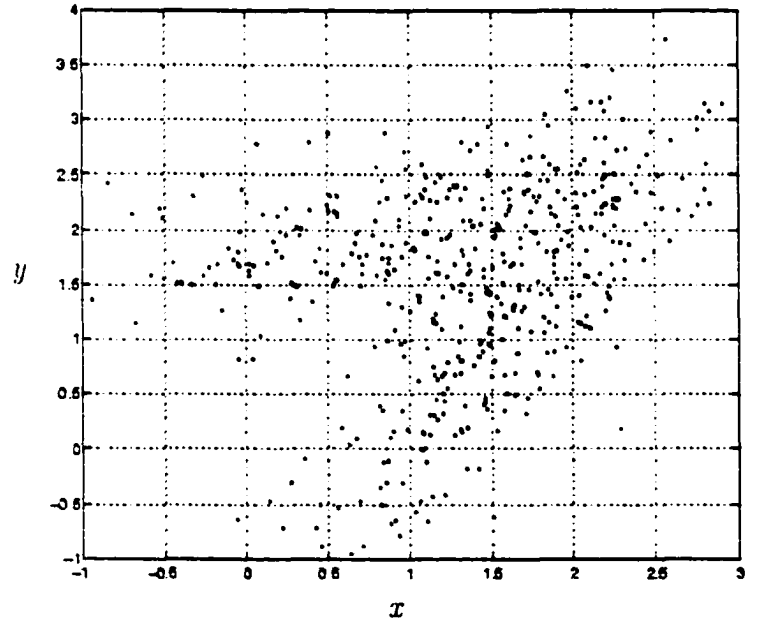


Figure 5.16: Samples from the V-density.

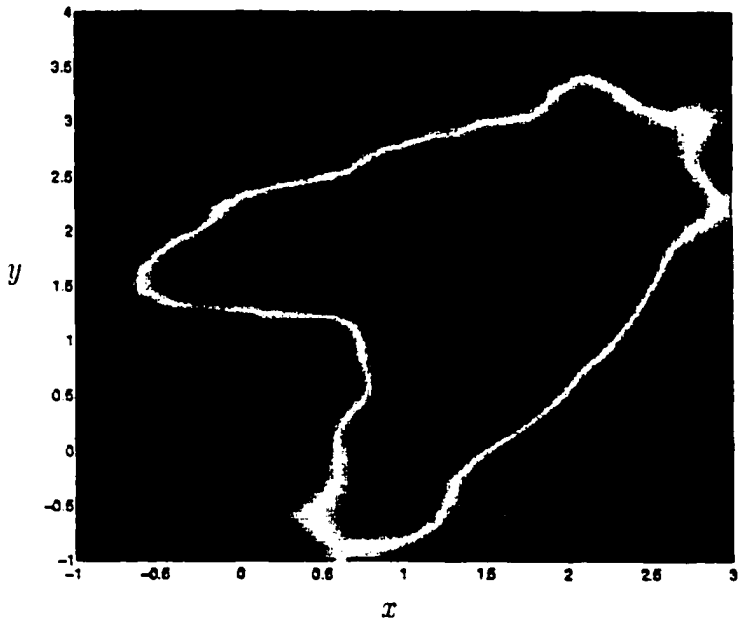


Figure 5.17: Kernel density estimate of the V-density with 500 data points.

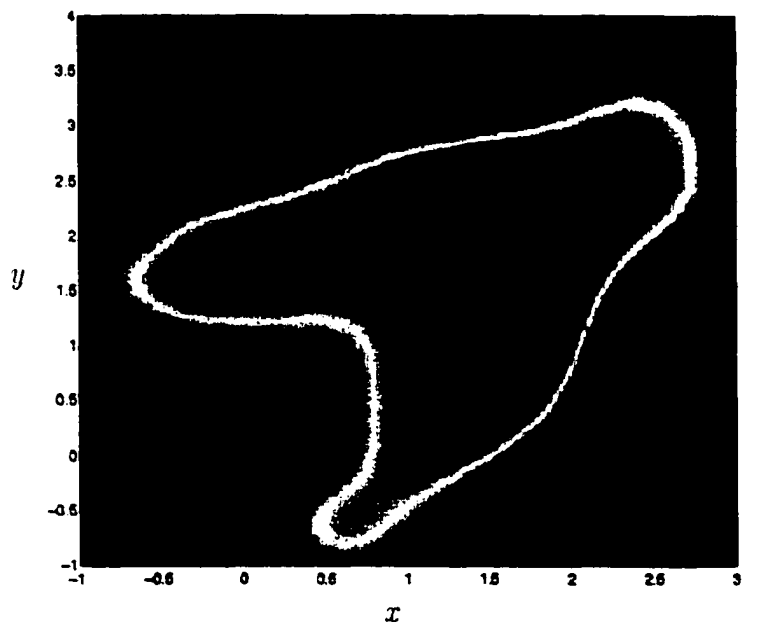


Figure 5.18: Maximum entropy density estimate of the V-density with 500 data points.

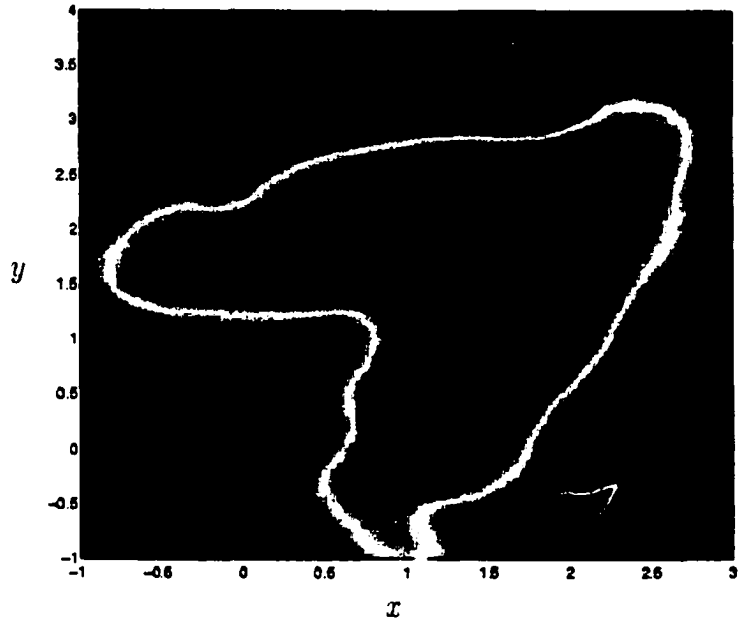


Figure 5.19: Kernel density estimate of the V-density with 1000 data points.

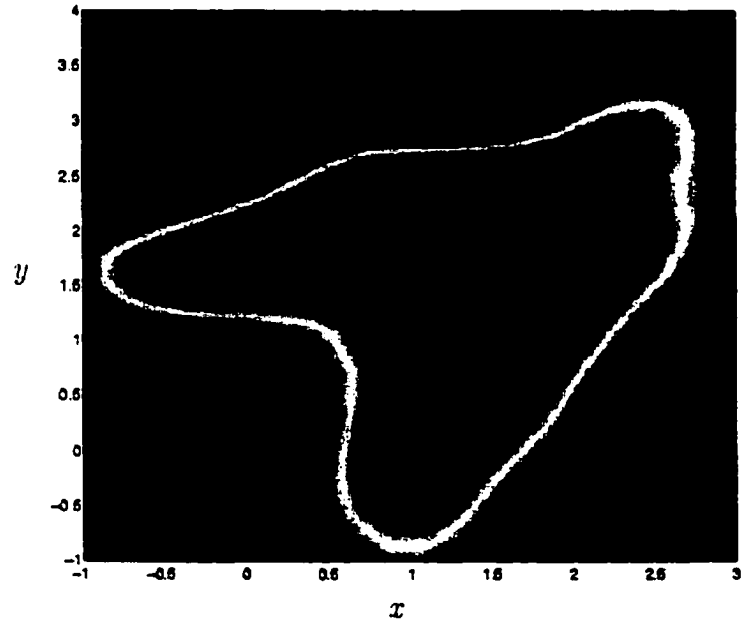


Figure 5.20: Maximum entropy density estimate of the V-density with 1000 data points.

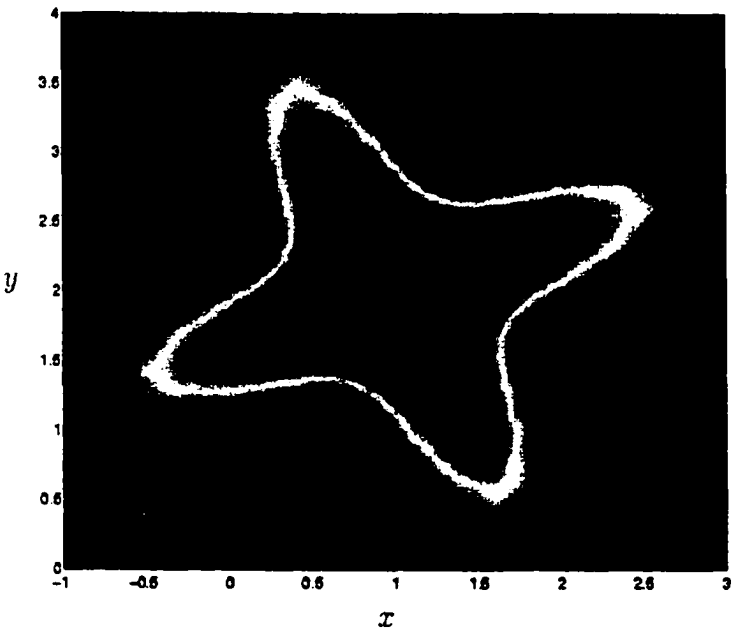


Figure 5.21: The theoretical density that we are taking our samples from. This is what we are trying to estimate. Because of its characteristic shape, we call it the X-density

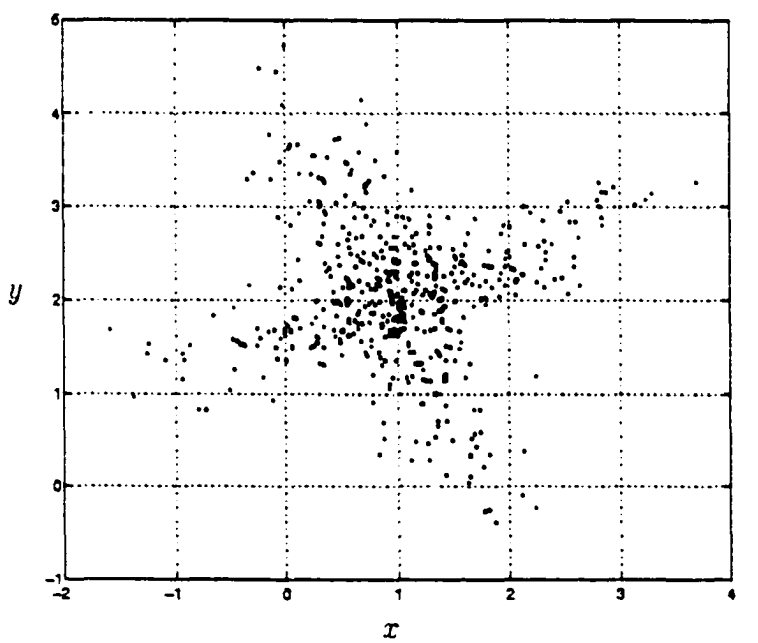


Figure 5.22: Samples from the X-density.

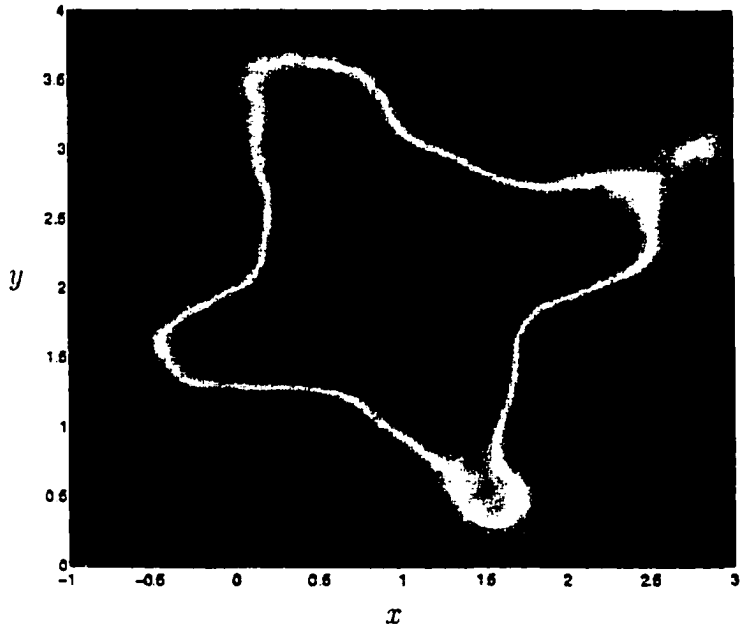


Figure 5.23: Kernel density estimate of the X-density with 500 data points.

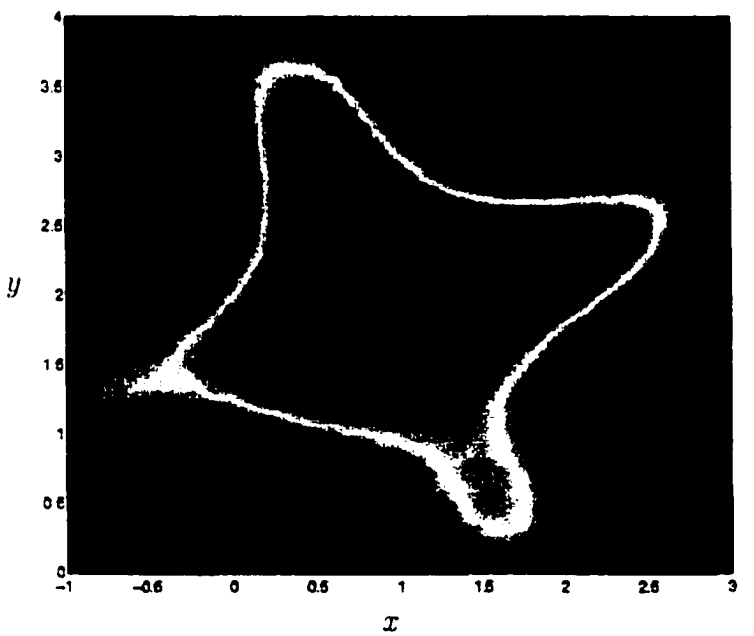


Figure 5.24: Maximum entropy density estimate of the X-density with 500 data points.

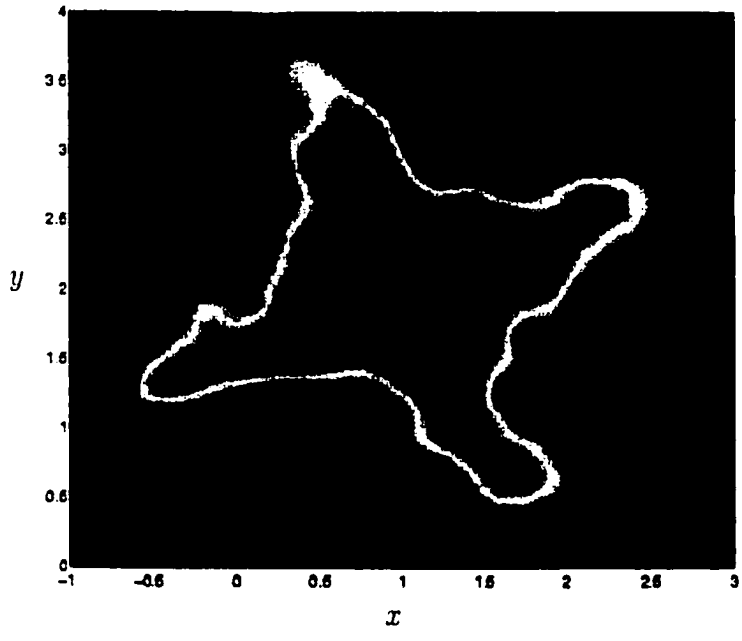


Figure 5.25: Kernel density estimate of the X-density with 1000 data points.

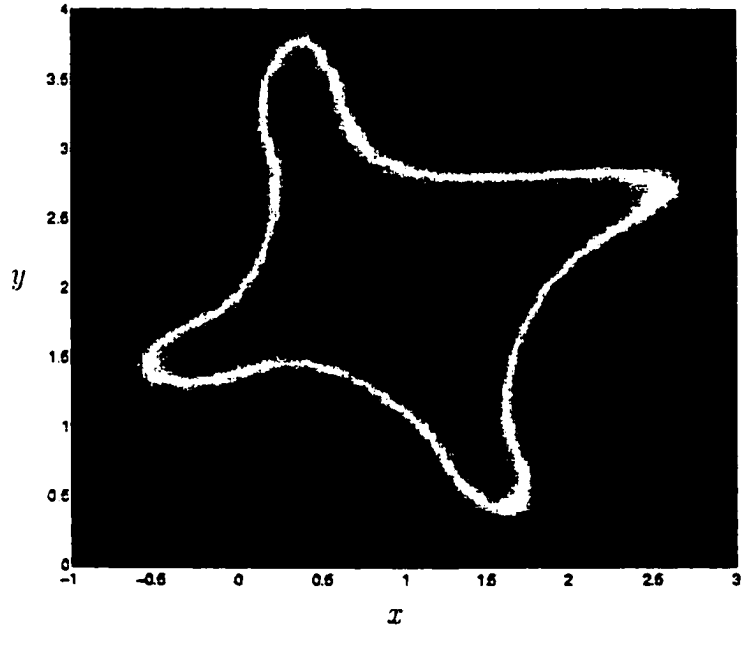


Figure 5.26: Maximum entropy density estimate of the X-density with 1000 data points.

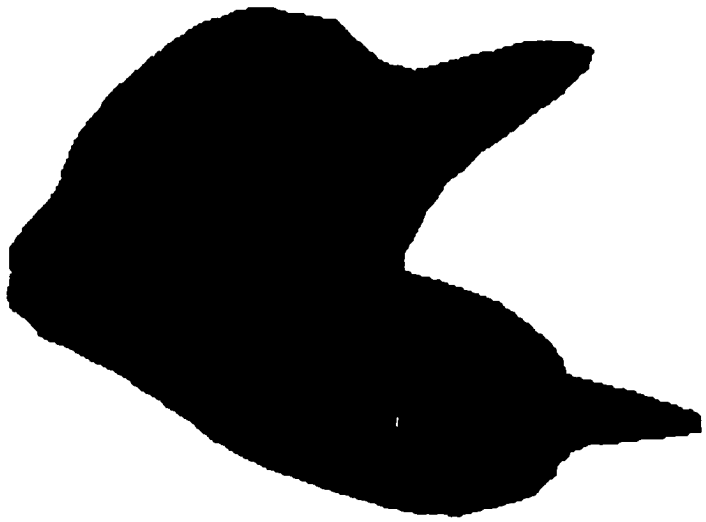


Figure 5.27: An equiprobable surface of the XV-density, the solution to $f_{XV}(\mathbf{x}) = 0.012$.

Figure 5.28: The same equiprobable surface as above, from a different angle.

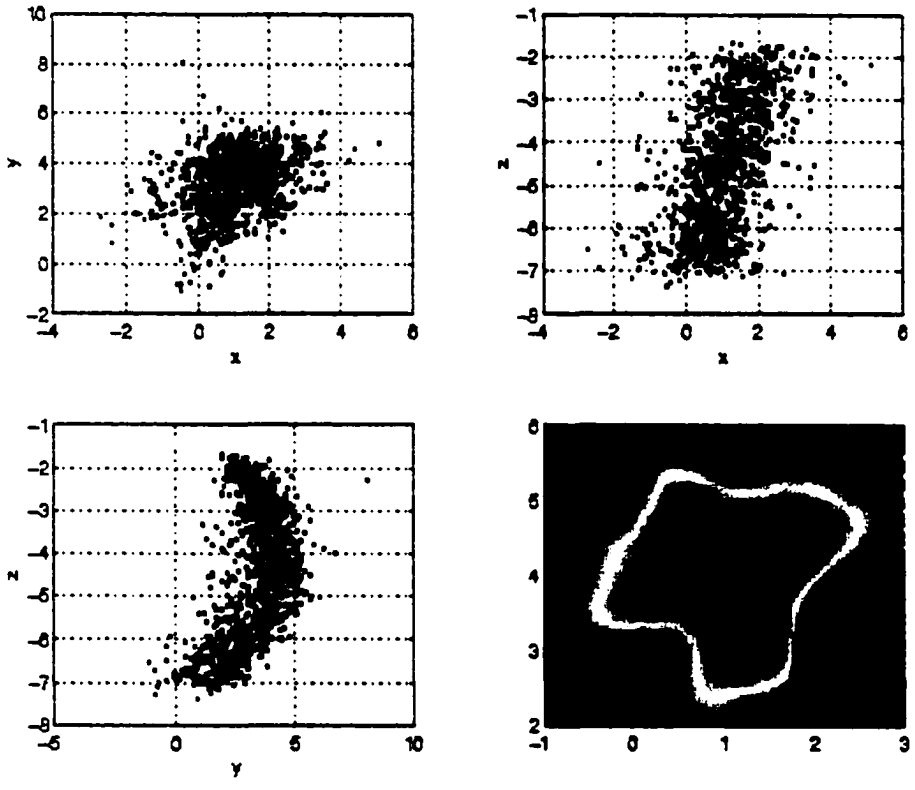


Figure 5.29: Data from the XV-density, plotted on two axes at a time. On the lower right is a cross section of the cylinder somewhere in the middle. The end cross sections are the well known X and V densities.

tance measure. Firstly, it is not really a distance measure, because the distance from $f(x)$ to $g(x)$ is not the same as the distance from $g(x)$ to $f(x)$. Secondly, if $g(x)$ is ever exactly zero over a piece of our domain, then our measure goes to infinity. In measuring the Kullback–Liebler distance, we must ensure that none of our densities go exactly to zero in our domain. We can do this by adding a tiny number (on the order of 10^{-10}) to each $g(x)$ that does go to zero.

Another metric of interest is the absolute error fraction between the original density and the estimate:

$$\text{absolute error fraction} = \int |f(x) - g(x)|\, dx \tag{5.33}$$

Sometimes in measuring the accuracy of the density estimates, we will find large errors, and wonder why we are not doing better. It is important to consider the performance of the density estimator in the context of the information it is given. Both the number of points the estimator is using to do its job, and the complexity of the underlying density should be taken into consideration.

Even an optimal density estimator, if given only a single data point taken from some underlying, multi–moded density, can do no more than assume some sort of spherical symmetry. There is simply no evidence in the data set for anything else, and thus no way for it to know that there are multiple modes. In such a case, there are not enough points to characterize the underlying density; we simply do not have enough information in our data set to do a good job. One might almost say that we need a preponderance of evidence in our data to lead us to believe that we have a bump in our density.

Our aim, then, is not to make the error go to zero because it might not be possible for that combination of density and data set. We simply aim to do as good a job as possible with the data we are given. Since we do not know the performance of an optimal density estimator, the best we can do is compare this technique to extisting techniques, and we chose to do this comparison with the most common existing technique (and one of the few multivariate ones), the kernel density estimator.

One feature that we have not quantified in the standard methods of measuring the error in our pdf estimates are the spurious high–frequency lobes that are sometimes introduced by the kernel density estimator, and most visible in Figure 5.25.

Computational Complexity

Once we have estimated our density function, the simplest operation we can perform on it is to evaluate it at a single point.

One of the advantages of the maximum entropy representation is that one can, to some extent, control the complexity of the representation by specifying the number of coefficients to use. This complexity of the representation is directly related the the complexity of evaluating the density at a point. The fewer coefficients, the more computationally efficient, and the sparser the representation. However, as the coefficients go down, so does the ability to represent details in the density.

We will compare the maximum entropy density estimate with the kernel density estimate in terms of computational efficiency in evaluating the density at a particular point. The fundamental unit of computational complexity that we use are multiply-adds.

We will assume that all computations are done in the most efficient way possible, using table lookups wherever possible to reduce floating point operations. We will also assume we are working with a multivariate data set in d dimensions that has N points in it. The maximum entropy estimator has c coefficients to represent the density in each dimension.

For each evaluation, the maximum entropy density has to find Legendre polynomials for each order from zero to $c-1$. These can be done by table lookup. It then incurs c multiply-adds per slice, and has to do c^{d-1} slices, for a total of c^d multiply-adds. It then has to repeat this process $d-1$ more times, as it collapses the multigrid. The total computational cost is simply $\sum_{i=0}^{d} c^i$ multiply adds.

On the other hand, the kernel density estimator only has to compute the distance of the point of interest from each of the data points. Each distance computation is d multiply-adds, and the total computational cost of a single evaluation is Nd multiply adds.

Only for N sufficiently high do we have a portion of our domain where we can construct a maximum entropy density that is more efficient than the kernel density estimator, namely when

$$\sum_{i=0}^{d} c^i < dN \tag{5.34}$$

At this point, it seems that the kernel density estimator is more computationally efficient, because it grows linearly with d while the maximum entropy technique grows exponentially with d. However, one must consider that c and N should be related,

and that c cannot be chosen arbitrarily using the maximum entropy technique.

Let us incorporate the relationship between c and N into our complexity measurement. To begin with, we should note that the number of coefficients in each dimension c should grow roughly proportional to the number of bins in that dimension. The reason for this is that adding a coefficient to a polynomial allows us to represent one more bump in a one dimensional curve. The one dimensional curve we are referring to is that produced by decomposition of a slice of bins from our sampled histogram. This is similar to the Nyquist sampling theorem, and thus we can relate how many bumps we should be trying to represent with our sample spacing. In addition to that, there should be some minimum number of points in each bin in the slice - otherwise, we have chosen an inappropriate bin size.

Thus, if there are approximately c non-zero bins in a single slice, and we require there to be at least p points in each bin, then there are cp points in each slice. By definition, there are c^{d-1} slices, and thus we can relate the total number of samples in the data set N to the number of coefficients: $pc^d \leq N$.

If we stipulate that there are on avaerage at least d points in each bin (the minimum number of points in a bin should go up with the dimensionality of the problem) then the relationship between samples of data and coefficients becomes: $c^{d+1} \leq N$.

The kernel density is more computationally efficient when it has fewer points in the data set, so despite our assumption that our data set should be larger than c^{d+1}, we will assume that it is equal to c^{d+1} exactly.

Substituting $N = c^{d+1}$ into Equation 5.34 yields:

$$\sum_{i=0}^{d} c^i < dc^{d+1} \tag{5.35}$$

When this inequality holds, we are guaranteed that the maximum entropy density estimator is more computationally efficient than the kernel density estimator.

The conclusions we can draw from these arguments are that if the data set we are trying to estimate a density for is very small and has high dimensionality (a case where one might think that the kernel density estimator would excel, computationally), there simply isn't enough information in the data set about the shape of the pdf to justify having a high value of c in the maximum entropy density estimate.

Thus, for many reasonable values of c (taking N and d into account), the maximum entropy density estimator is more computationally efficient than the kernel density estimator. As we will see in Table 5.3, the maximum-entropy density estimate is two to seven times more computationally efficient than the kernel density estimate.

Sparse Representation

As mentioned above, controlling the number of coefficients c in the maximum entropy estimator controls the spaseness of the representation. This may be important for storing the density estimates, if there are many of them. The kernel density estimator uses the whole data set as representation, which for a high dimensional data set with many points, can be unwieldly. This comparison is very similar the the one presented in the preceding section. In this case, the unit of storage is a double precision floating point number.

The maximum entropy estimate has c^d double precision coefficients to represent it, while the kernel density estimator has Nd coefficients that represent it. The maximum entropy representation is sparser when

$$c^d < dN \tag{5.36}$$

Using what we found in the computational efficiency section, we can use the relationship we found between c and N. Substituting $N = c^{d+1}$ into the efficiency comparison yields the following condition for the maximum entropy density estimator to have a sparser representation:

$$c^d < dc^{d+1} \tag{5.37}$$

Thus, we can state that the maximum entropy density representation under these conditions is always sparser than the kernel density estimator. Note that just as in the computational efficiency case, Table 5.3 of results indicates that the maximum entropy density is two to seven times more computationally efficient than the kernel density estimator.

Numerical Results

We present numerical results of the estimates that we made, using the synthetically generated known densities described earlier in Section 5.5.

While reading Table 5.3, we would do well to keep in mind that the accuracy of our estimate is affected and limited by many variables, including the dimensionality of our data, the shape of the underlying density, and the number of points we are using to do our estimate. The M–A column is the number of multiply adds it takes

to compute the density at a particular point, and the Stor column is the number of double precision storage units it takes to represent the density (essentially this is a measurement of sparseness of the representation).

	Estimator	N	$\int \lvert f-g \rvert dx$	$D(f \parallel g)$	M-A	Stor
V-density	MaxEnt 15 0.6/0.6	500	0.1968	0.0367	240	225
	Kernel 0.6		0.1832	0.0462	1,000	1,000
	MaxEnt 15 0.6/0.6	1,000	0.1678	0.0330	240	225
	Kernel 0.5		0.1685	0.0370	1,000	1,000
X-density	MaxEnt 15 0.7/0.7	500	0.2510	0.0570	240	225
	Kernel 0.5		0.2315	0.0621	1,000	1,000
	MaxEnt 15 0.45/0.45	1,000	0.1937	0.0369	240	225
	Kernel 0.4		0.1470	0.0376	1,000	1,000
XV-density	MaxEnt 8 1.1/1.1/1.1	1,000	0.6284	0.3958	584	512
	MaxEnt 10 1.1/1.1/1.1		0.5749	0.3643	1,110	1,000
	Kernel 1.1		0.5845	0.5088	3,000	3,000
	MaxEnt 12 1.1/1.1/1.1	5,000	0.5826	0.3274	1,884	1,726
	Kernel 1.0		0.5584	0.3354	15,000	15,000

Table 5.3: Accuracy and efficiency of the Maximum Entropy Density Estimation

5.6 Conclusions

Now, taking a look back at what we have done, we have gone from a set of data to a grid of points in multidimensional space to a grid of coefficients. What have we gained?

- We have found a smooth, analytical, scalar valued probability density function that is very likely close to the one that created the samples.
- By using the functional form that we have, we have ensured that this is the maximum entropy function that matches our constraints. This is a theoretically sound, defensible procedure.
- Compared to the kernel density estimator, we have in many cases significantly reduced the number of coefficients, achieving a much sparser representation than before.
- Compared to the kernel density estimator, the maximum entropy representation is, in many cases, much more computationally efficient to evaluate.
- Compared to the kernel density estimator, the maximum entropy representation does not introduce spurious high frequency components into the density being estimated.
- This whole procedure works for multivariate (multidimensional) densities with an arbitrary number of points in their data sets.

- Depending on storage or computational requirements, you can trade off fidelity and efficiency (storage and computational) of the representation by selecting different numbers of coefficients to use in the representation.

CHAPTER 6

Classification of Short Vegetation Using Polarimetric, Multifrequency SAR

6.1 Bayesian-Hierarchical Classification

Even after we have a solid technique for estimating densities, practical problems remain in applying it to our data and performing an optimal Bayesian classification.

One of the problems that we encounter when estimating probability density functions is that in higher dimensions, we require a great deal of data to characterize our density. In fact, as the number of dimensions goes up, the amount of data needed to characterize a density increases roughly exponentially.[1] If our measurements are eight dimensional (as they are in our case), we need an astronomical number of training points to keep our estimate accuracy high.

To address this problem, we noted that the higher correlations required to separate wheat from everything else are different than those needed to separate corn from everything else, and so on for each different vegetation class. Each type of vegetation can be distinguished from the rest by using a different subset of correlations. Rather than trying use all the correlations at once, as the Bayesian technique does, we can estimate a few at a time, and perform the classification in a Hierarchical manner, identifying each class, one at a time.

We identified sets of correlations between measurement channels that were useful for classification. Then, for each class of interest (corn, soybeans, wheat[2], alfalfa, bare) we produced probability density functions that contained those correlations.

We did this by gouping appropriate measurement channels together, estimating the pdf of each group of channels, and taking the product of the resulting marginal

[1] Because the "volume" increases exponentially with dimension.

[2] For the wheat vegetation class, we split the data set into two groups, finding pdfs of early and late wheat seperately. We did the same for short and tall alfalfa.

probability density functions to find the overall density.

At this point, we applied the pdfs and produced a Bayesian classifier for each different set of correlations.

We used the pdfs produced by the Bayesian classifiers to define decision boundaries, analogous to the ones we drew by hand in Section 4.2. Using pdfs to define decision boundaries freed us from the manual labor of drawing them by hand. It also allowed us to define these boundaries in higher dimensions, thus exploiting the higher dimensional correlations that came along with them.

For each data point, we could then look at the result from each classifier and make a decision that was based on the classifiers that had correlations important to seperating the class of interest.

In this way, we combined ideas from the Hierarchical classifier and optimal Bayesian classifier, to produce what we call the Bayesian-Hierarchical classifier.

6.1.1 Details of Applying the Bayesian-Hierarchical Technique to Classification Short Vegetation

In the case of classification of short vegetation, we computed eight different pdfs, each comprised of a different combination of marginal densities. The pdfs and their marginals are given in Table 6.1. The subscript (A-H) on each pdf identifies the particular combination of marginals, and thus the corresponding combination of correlations that were estimated. These subscripts also identify the corresponding Bayesian classifiers that we produce with those pdfs.

$f_A(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\sigma^o_{LHH}, \sigma^o_{LHV}, \sigma^o_{CHH}) \cdot f(\sigma^o_{LVV}, \sigma^o_{CHV}, \alpha_C) \cdot f(\alpha_L, \sigma^o_{CVV})$
$f_B(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\alpha_L, \sigma^o_{CVV}, \alpha_C) \cdot f(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{CHH}) \cdot f(\sigma^o_{LHV}, \sigma^o_{CHV})$
$f_C(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\sigma^o_{LHH}, \alpha_L, \sigma^o_{CHH}) \cdot f(\sigma^o_{LVV}, \sigma^o_{CHV}, \alpha_C) \cdot f(\sigma^o_{LHV}, \sigma^o_{CVV})$
$f_D(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}) \cdot f(\alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}) \cdot f(\sigma^o_{CHV}, \alpha_C)$
$f_E(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\sigma^o_{LVV}, \sigma^o_{LHV}, \sigma^o_{CVV}) \cdot f(\sigma^o_{LHH}, \alpha_L, \sigma^o_{CHH}) \cdot f(\sigma^o_{CHV}, \alpha_C)$
$f_F(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{CHV}) \cdot f(\sigma^o_{LHV}, \alpha_L, \alpha_C) \cdot f(\sigma^o_{CHH}, \sigma^o_{CVV})$
$f_G(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\sigma^o_{LVV}, \alpha_L, \sigma^o_{CHH}) \cdot f(\sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) \cdot f(\sigma^o_{LHH}, \sigma^o_{LHV})$
$f_H(\sigma^o_{LHH}, \sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L, \sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}, \alpha_C) =$ $f(\sigma^o_{LVV}, \sigma^o_{LHV}, \alpha_L) \cdot f(\sigma^o_{CHH}, \sigma^o_{CVV}, \sigma^o_{CHV}) \cdot f(\sigma^o_{LHH}, \alpha_C)$

Table 6.1: Probability density functions (pdfs) used for classification of short vegetation. Each probability density function shown is the product of three marginal pdfs. Each set of marginals corresponds to a different set of correlations that are being estimated. We use each of these combinations of correlations to produce a Bayesian classifier.

Each Bayesian classifier partitions the set of data we intend to classify. We will denote the partitioned subsets with the name of the Bayesian classifier that created them.

For example, Bayesian classifier A partitions our data set into five subsets:

$$A_{\text{alfalfa}}, A_{\text{bare}}, A_{\text{corn}}, A_{\text{soy}}, A_{\text{wheat}} \tag{6.1}$$

The set A_{wheat} is the set of measured data that classifier A identified as wheat.

We will also define the sets

$$R_{\text{alfalfa}}, R_{\text{bare}}, R_{\text{corn}}, R_{\text{soy}}, R_{\text{wheat}} \tag{6.2}$$

as initially empty. These sets will contain the final results of our classification.

Tables 6.2 and 6.3 list the combinations of decision rules used by the Bayesian-Hierarchical classifier to classify short vegetation. The notation used for each step consists of set operations on the partitions produced by our Bayesian classifiers, the result of which is added into the set indicated to the right of the arrow.

After each step that is shown, the data points added into the final (R) sets are removed from consideration during the rest of the classification procedure. For clarity of presentation, these removal steps are omitted from the Tables Tables 6.2 and 6.3.

6.2 Results

In this section we will carefully evaluate the Bayesian-Hierarchical classifier using a variety of different measures of accuracy.

There are three main measures of accuracy we consider: accuracy from the users

1. $A_{\text{corn}} \cap B_{\text{corn}} \Longrightarrow R_{\text{corn}}$

2. $C_{\text{corn}} \cap D_{\text{corn}} \cap E_{\text{corn}} \cap F_{\text{corn}} \cap G_{\text{corn}} \Longrightarrow R_{\text{corn}}$

3. $A_{\text{soy}} \cap B_{\text{soy}} \Longrightarrow R_{\text{soy}}$

4. $C_{\text{soy}} \cap D_{\text{soy}} \Longrightarrow R_{\text{soy}}$

5. $A_{\text{tall alfalfa}} \cap B_{\text{tall alfalfa}} \cap C_{\text{tall alfalfa}} \cap E_{\text{tall alfalfa}} \cap F_{\text{tall alfalfa}} \Longrightarrow R_{\text{alfalfa}}$

6. $A_{\text{tall alfalfa}} \cap B_{\text{tall alfalfa}} \cap D_{\text{tall alfalfa}} \cap F_{\text{tall alfalfa}} \Longrightarrow R_{\text{alfalfa}}$

7. $A_{\text{early wheat}} \cap B_{\text{early wheat}} \cap C_{\text{early wheat}} \cap H_{\text{early wheat}} \cap E_{\text{early wheat}} \Longrightarrow R_{\text{wheat}}$

8. $C_{\text{early wheat}} \cap G_{\text{early wheat}} \cap H_{\text{early wheat}} \Longrightarrow R_{\text{wheat}}$

9. $D_{\text{early wheat}} \cap G_{\text{early wheat}} \cap H_{\text{early wheat}} \Longrightarrow R_{\text{wheat}}$

10. $B_{\text{early wheat}} \cap E_{\text{early wheat}} \cap G_{\text{early wheat}} \cap H_{\text{early wheat}} \Longrightarrow R_{\text{wheat}}$

11. $A_{\text{late wheat}} \cap B_{\text{late wheat}} \cap C_{\text{late wheat}} \cap D_{\text{late wheat}} \cap E_{\text{late wheat}} \cap H_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

12. $C_{\text{late wheat}} \cap G_{\text{late wheat}} \cap H_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

Table 6.2: First twelve decision rules used by the Bayesian-Hierarchical classifier for short vegetation. Each set comes from the Bayesian classifier produced from the pdf of Table 6.1 with the same subscript. Thus A_{corn} is the set of data points that the Bayesian classifier produced from the marginals in Table 6.1 identified as corn.

13. $A_{\text{late wheat}} \cap C_{\text{late wheat}} \cap D_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

14. $A_{\text{late wheat}} \cap F_{\text{late wheat}} \cap H_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

15. $A_{\text{late wheat}} \cap C_{\text{late wheat}} \cap E_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

16. $B_{\text{late wheat}} \cap C_{\text{late wheat}} \cap H_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

17. $A_{\text{late wheat}} \cap B_{\text{late wheat}} \cap C_{\text{late wheat}} \cap D_{\text{late wheat}} \cap F_{\text{late wheat}} \cap H_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

18. $B_{\text{late wheat}} \cap C_{\text{late wheat}} \cap F_{\text{late wheat}} \cap G_{\text{late wheat}} \cap H_{\text{late wheat}} \Longrightarrow R_{\text{wheat}}$

19. $A_{\text{alfalfa}} \cap C_{\text{alfalfa}} \cap D_{\text{alfalfa}} \cap E_{\text{alfalfa}} \Longrightarrow R_{\text{alfalfa}}$

20. $D_{\text{alfalfa}} \cap H_{\text{alfalfa}} \Longrightarrow R_{\text{alfalfa}}$

21. $A_{\text{alfalfa}} \cap D_{\text{alfalfa}} \Longrightarrow R_{\text{alfalfa}}$

22. $A_{\text{bare}} \cup B_{\text{bare}} \cup E_{\text{bare}} \cup F_{\text{bare}} \cup G_{\text{bare}} \cup H_{\text{bare}} \Longrightarrow R_{\text{bare}}$

23. Remaining data $\Longrightarrow R_{\text{wheat}}$

Table 6.3: Last eleven decision rules used by the Bayesian-Hierarchical classifier for short vegetation. Each set comes from the Bayesian classifier produced from the pdf of Table 6.1 with the same subscript. Thus A_{corn} is the set of data points that the Bayesian classifier produced from the marginals in Table 6.1 identified as corn.

perspective, accuracy from the producers perspective and overall accuracy.

The producer accuracy for class 'C' measures what percent of the data from class 'C' was correctly classified. The user accuracy for class 'C' measures what percent of the data identified as class 'C' indeed belongs to class 'C.' The overall accuracy is what percent of all the data was correctly classified.

For each evaluation of accuracy for the Bayesian-Hierarchical classifier, we divided our data set into two sets: a training data set, and a testing data set. We used the training data set to estimate the densities involved in our classification, and we used the testing data set as an independant data set with which to evaluate the performance of the classifier.

To begin our evaluation of classifier accuracy, we present Table 6.4, a confusion matrix evaluating the performance of the Bayesian-Hierarchical classifier for the data set we have available. Note that the red number at the top of the confusion matrices is the averaged producer accuracy over all the classes, and gives a single metric to tell us how well we are doing.

In order to offer a clear picture of how the Bayesian-Hierarchical technique compares to other techniques, we classified the same data with ISODATA (Table 6.5), and with a MLE classifier which assumes Gaussian densities for the statistics of each class (Table 6.6.)

With these confusion matrices, we observe that the Bayesian-Hierarchical classifier improves on the average producer accuracies of the other classes by 10 and 20%.

Bayesian-Hierarchical 95%

	Classified As				
True Class	Alfalfa	Bare	Corn	Soy	Wheat
Alfalfa	92	3	0	2	3
Bare	2	95	2	0	0
Corn	0	3	97	0	1
Soy	4	0	0	96	0
Wheat	2	2	0	2	94

Table 6.4: A confusion matrix for short vegetation produced with the Bayesian-Hierarchical classifier. The data set it was produced with spans an entire growing season.

ISODATA 74%

	Classified As				
True Class	Alfalfa	Bare	Corn	Soy	Wheat
Alfalfa	62	8	0	2	28
Bare	10	73	7	0	10
Corn	1	16	81	0	3
Soy	4	1	0	74	22
Wheat	15	1	0	1	82

Table 6.5: A confusion matrix for short vegetation produced with the ISODATA unsupervised classifier. The data set it was produced with spans an entire growing season.

MLE with Gaussian Assumption 84%

	Classified As				
True Class	Alfalfa	Bare	Corn	Soy	Wheat
Alfalfa	76	2	2	5	16
Bare	9	68	17	0	7
Corn	0	6	92	0	2
Soy	3	0	0	96	0
Wheat	9	2	0	0	88

Table 6.6: A confusion matrix for short vegetation produced with the MLE classifier with Gaussian assumptions. The data set it was produced with spans an entire growing season.

6.2.1 Other Measures of Classifier Accuracy

Because the results of the classifier depend heavily on the data used to train it, we randomized the selection of training and testing sets of data. The whole classification, from training and density estimation to classification of the independent data set was performed one hundred times, and the accuracies of each trial were averaged.

The results of these trials are pictured in Figures 6.1-6.6, for different amounts of data used for training and testing.

This evaluates user and producer accuracy over many instances of the classifier and also measures the sensitivity of the density estimation technique to variations in the size of the data set used for density estimation.

Note in Figure 6.4 that the user accuracies for soybeans tend to be low only because of the relative size of the soybean data set. If data sets were of comparable size, the soybean user accuracy would likely be almost 96% rather than the relatively low 86% that we computed.

6.2.2 Image Domain

Another evaluation of the accuracy was performed in the image domain. We took one of the radar images, and classified it using the Bayesian-Hierarchical method and the densities that we had estimated before. We then classified it using ISODATA, an unsupervised clustering technique commonly found in the literature. The ground truth and the resulting classifications are compared in Figures 6.7-6.9.

Although neither classification is perfect, the reader will note the relative lack of

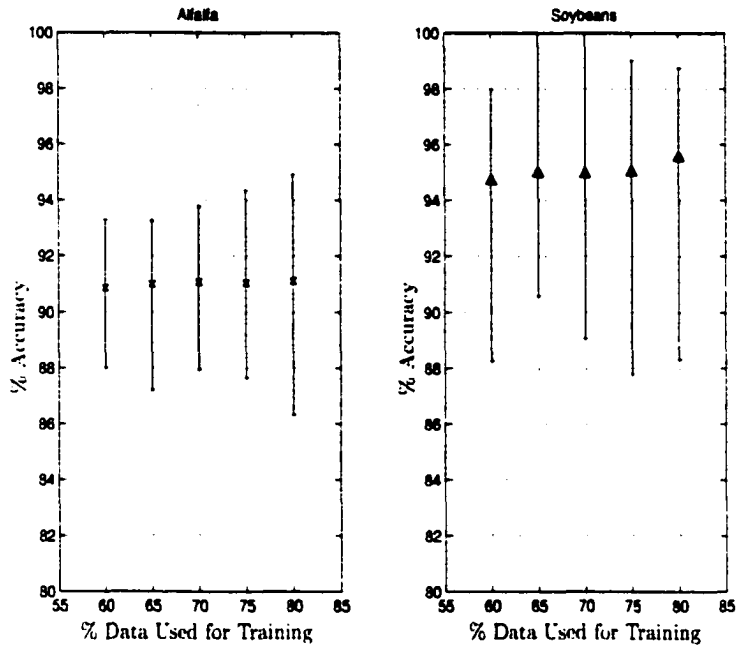


Figure 6.1: The producer accuracy for alfalfa and soybeans. The horizontal axis is the percent of data used for training, and the vertical axis is the producer accuracy for that class. The indicator shows the average producer accuracy over 100 instances of the classifier, while the error bars indicate the maximum and minimum accuracies.

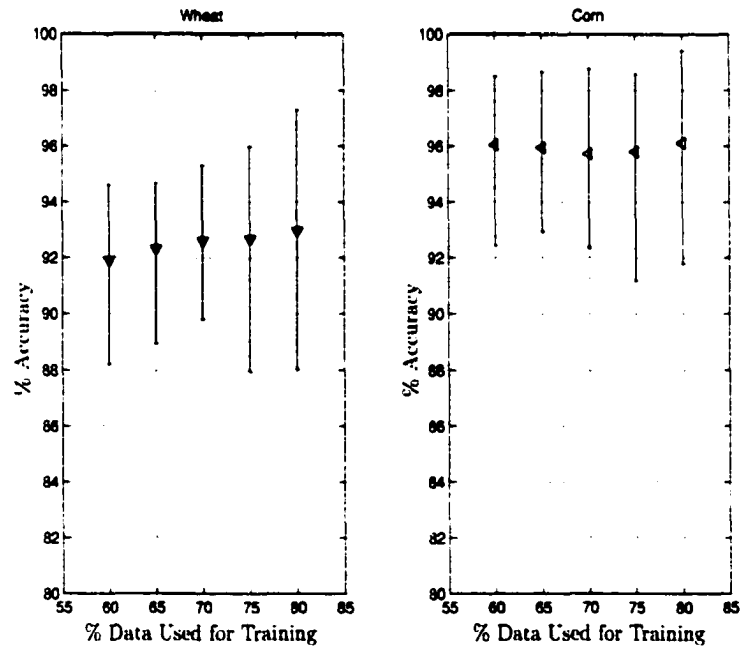


Figure 6.2: The producer accuracy for the wheat and corn classes. As before, the horizontal axis is the percent of data used for training, and the vertical axis is the producer accuracy for that class.

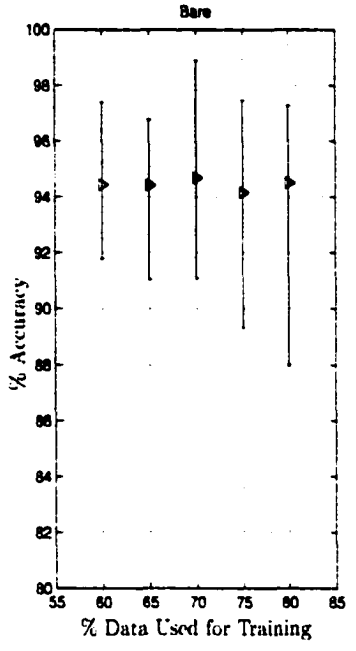


Figure 6.3: The producer accuracy for the bare class. As before, the horizontal axis is the percent of data used for training, and the vertical axis is the producer accuracy for that class.

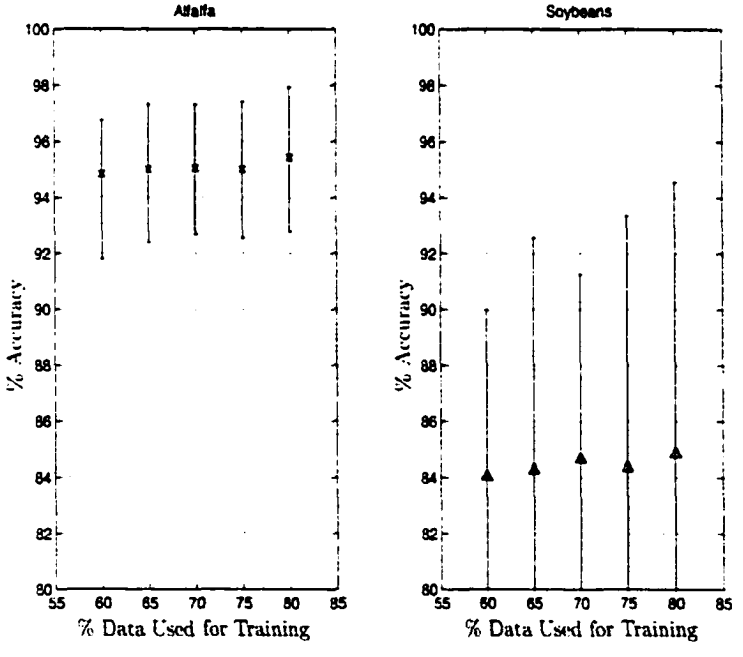


Figure 6.4: The user accuracy for alfalfa and soybeans. The horizontal axis is the percent of data used for training, and the vertical axis is the user accuracy for that class. The indicator shows the average user accuracy over 100 instances of the classifier, while the error bars indicate the maximum and minimum accuracies.

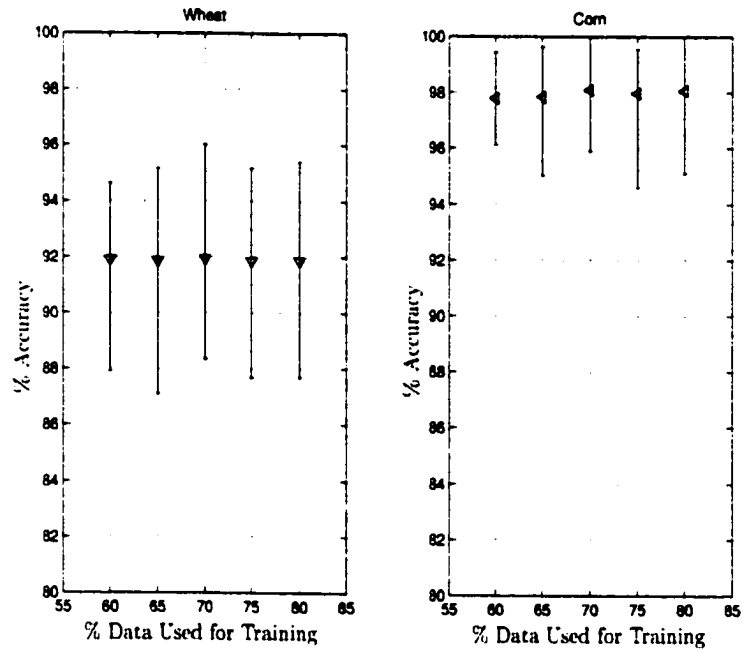


Figure 6.5: The user accuracy for the wheat and corn classes. As before, the horizontal axis is the percent of data used for training, and the vertical axis is the user accuracy for that class.

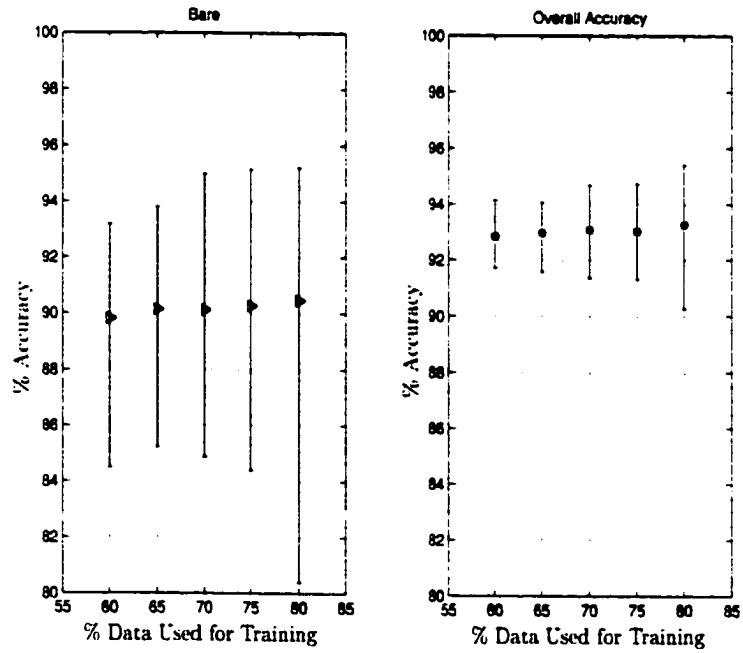


Figure 6.6: The user accuracy for bare is plotted on the left, while the overall accuracy is plotted on the right. As before, the horizontal axis is the percent of data used for training, and the vertical axes are the measured accuracies.

accuracy in the ISODATA classification, which confuses wheat with alfalfa in many instances in the lower right corner of the image.

6.3 Conclusions

Using the Bayesian-Hierarchical classifier to identify short vegetation gives us an overall accuracy of around 93%.

We have tested it with one of the larger data sets in the literature, which includes many fields over the entire growing season of the vegetation. A data set that large allows us to test the classifier over most of the range of biophysical conditions that the vegetation might be in.

In addition to the size, the training and testing data sets were independent, and this is an important factor in evaluating real-world performance. Because many classifiers are tested on small data sets, frequently the researcher has no choice but to use the same data set for training and testing. This gives results that are somewhat biased, and show the classifier performing better than it really is capable of.

This classification technique works because we are using higher order correlations between different channels. Very few techniques do this or have the potential to exploit the correlations the way the Bayesian-Hierarchical technique does. This is really what sets this technique apart from other classification methods in the literature.

This technique is characterized by the following features:

- High accuracy
- Excellent computational efficiency, due to its use of the maximum-entropy den-

Figure 6.7: Ground-truth information indicating vegetation type that we know to have been growing on the ground in the summer of 1995 on this particular date.

Figure 6.8: A classified image of Kellog Biological Station made using ISODATA.

Figure 6.9: A classified image of Kellog Biological Station made using the Bayesian-Hierarchical classifier.

sity estimation technique.

- Ease of application to identifying regions where classes overlap.
- Adaptation to the addition of additional classes without retraining

The main disadvantage of the technique is the large amount of data initially needed in the training phase.

CHAPTER 7

Applications of a Statistical Framework to Other Remote Sensing Problems

7.1 Introduction

In the preceding two chapters, we have taken a statistical approcah to classification. Rather than attempting to develop a representation of electromagnetic interactions with vegetation from first principles (i.e. Maxwells equations) we have chosen to represent the statistics of our vegetation measurements, using multidimensional probability density functions and classical statistical methods.

Although our previous efforts focused on developing an algorithm with which to classify short vegetation, the inquisitive reader might wonder if the statistical techniques we developed could be put to use in solving other problems or for other applications.

As a model, the statistical approach has some advantages when compared to standard electromagnetic models. Unlike standard techniques, these conclusions do not have a limited region of validity; they are valid where the model says they are, because the "model" is simply a probability density function that comes directly from the data itself.

Of course, the statistical representation has disadvantages, also. For example, it cannot ever tell us specific information about the electromagnetic scattering mechanisms involved, for example whether corn's strong response is really from a double bounce mechanism off the stalk.

Thus, the relative usefulness of the statistical approach depends on what questions we would like to answer. We will consider two cases that benefit from the application of the statistical techniques.

- First, we will examine the possibility of representing the non-stationarity of the vegetation statistics. We will call this technique parameterizing the probability density function, because that is a good description of what we are doing: we are constructing a new probability density function of the vegetation statistics, which is conditioned on some of the biophysical parameters.

  One application of this representation could be the estimation of biophysical parameters of short vegetation, based on the electromagnetic response from that area of the ground. The parameters we can estimate are those that affect the measurements, such as plant height and plant biomass. We sometimes call these *input parameters,* and they are usually part of our ground truth.

  This estimation process can only be done once the vegetation has been properly classified; i.e. we cannot tell what the height of a particular plant is unless we know what kind of plant it is.

- Second, we will apply our statistical techniques to help us improve our ability to detect point targets in a vegetation background.

  We will do this in two ways: The first is an extension of what we did for vegetation, simply using our density estimator to represent the higher order correlations in a multidimensional measurement. The second is to incorporate information about the vegetation statistics (and non-stationarity) into our existing representations of point targets, such as the Nakagami-Rice distribution [39].

Both parameter estimation and point target detection are tasks which benefit from

the application of the statistical methods developed in the previous chapters. In the following discussion, we examine these applications in more depth. This treatment is meant to provide demonstration of these concepts, rather than a full blown analysis and application of these approaches.

7.2 Representation of Non–Stationary Densities

The first task we will examine is the representation of the non–stationarity of the vegetation statistics. Our intention here is to have the model statistics change over time, reflecting changes in the plant height, biomass, soil moisture, and other relevant parameters.

As we have noted before, the measurements of scattering matrix elements for a single pixel of vegetation with fixed biophysical parameters have Gaussian fading characteristics.[1]

Measurements of scattering matrix elements for a single vegetation type are not Gaussian because of the fluctuations in the biophysical parameters (height, biomass, soil moisture, planting denisty) that go on over time and in different fields.

This change in biophysical parameters causes a change in statistical character over time and space and is the basis for the non–stationarity behavior of our vegetation classes. Another way to look at what we are trying to do is to say that we are representing a class of vegetation using a covariance matrix by allowing the elements

[1] The statistics of a vegatation pixel with fixed inputs was discussed in Section 2.2.3 and illustrated in Figure 2.5.

in the covariance matrix to have their own statistical character.[2]

We would like to find this statistical character when it is conditioned on one of the input parameters. This would allow the density that we use to describe a particular class to change and shift around in our multidimensional measurement space as the input parameter of interest changed.

We will call this procedure parameterization of the probability density function, and we should note that to do it correctly, we need a data set with sufficient ground truth measurements of the parameter of interest.

7.2.1 Parameterized Probability Density Functions

In order to develop this representation, and parameterize the density function, we will need to estimate the probability density function of a particular class, when it is conditioned on a particular value of one of its input parameters. All of the other input parameters are varying in the normal fashion, and because of our lack of knowledge about them, they are lending uncertainty to our measurements.

To illustrate, consider the following more abstract example, where we have four inputs for each pixel, S_0, S_1, S_2, S_3. The first input represents target type; let us fix this for the time being. Now fix another input, say $S_1 = a'$, and let the others vary throughout the range of their possibilities. We can then estimate $f_{S_2,S_3}(\boldsymbol{M} = \boldsymbol{m}|S_0, S_1 = a')$ using the procedure outlined in Chapter 5. Now, set $S_1 = a''$, still

[2]We should note that this discussion simplifies things somewhat, since we aren't actually using covariance matrix elements, but actually working with radar backscattering coefficients and phase statistics. To go from one representation to the other is simply a transformation, and thus does not fundamentally affect our discussion or conclusions at this point. We could more accurately say that we are allowing the radar bacscattering coefficients and co-polarized alpha phase statistic to have their own statistical character.

holding the target (S_0) fixed, and then let the other inputs vary through their domain of possibilities. We estimate another pdf, $f_{S_2,S_3}(\boldsymbol{M} = \boldsymbol{m}|S_0, S_1 = a'')$, again using the procedure in Chapter 5. The subscripts on the functions indicates that the uncertainty in these pdfs come from lack of knowledge about the input S_2, S_3.

Interpolating between the two densities, we can effectively find $f(\boldsymbol{M} = \boldsymbol{m}|S_0, S_1 = a)$ for all inputs a between a' and a''. In addition, this type of representation can be easily inverted to do parameter estimation.

We can do a linear interpolation thus

$$\begin{aligned} f_{S_2,S_3}\left(\boldsymbol{M} = \boldsymbol{m} - \left\{\left(\frac{a-a''}{a'-a''}\right) E\left[\boldsymbol{M}|S_0, S_1 = a'\right] + \right.\right. \\ \left.\left.\left(\frac{a'-a}{a'-a''}\right) E\left[\boldsymbol{M}|S_0, S_1 = a''\right]\right\} |S_0, S_1 = a\right) = \\ \left(\frac{a-a''}{a'-a''}\right) f_{S_2,S_3}\left(\boldsymbol{M} = \boldsymbol{m} - E\left[\boldsymbol{M}|S_0, S_1 = a'\right]|S_0, S_1 = a'\right) + \\ \left(\frac{a'-a}{a'-a''}\right) f_{S_2,S_3}\left(\boldsymbol{M} = \boldsymbol{m} - E\left[\boldsymbol{M}|S_0, S_1 = a''\right]|S_0, S_1 = a'\right) \end{aligned} \tag{7.1}$$

Although the above equation is complicated to write down, the concept is simple. It is very literally a linear interpolation of the motion of the mean of the density, and a simultaneous linear interpolation of the corresponding densities.

Each target could then be represented as a pdf in measurement space that moves and changes shape as the target inputs change. If we have enough data, we can parameterize multiple inputs, representing each with its own parameter. We can also study how one input changes the target measurements without completely understanding how the other inputs affect the outputs. This is simply a way to encapsulate and represent as much of the information about a target as we have available.

	5/31, 6/2, 6/4	7/10, 7/12, 7/14
A	66	94
B	62	91
C	66	94
D	62	90
E	62	97
F	59	103
G	50	87
H	49	89
	measured on 6/1	measured on 7/11

Table 7.1: Ground truth measurements of the height of wheat during the 1995 growing season, for a set of 8 fields.

7.2.2 Non–Stationary Densities of Wheat Measurements

Let us apply the parameterized density technique to a real world example. We chose to parameterize the wheat density by the height of the vegetation. The end product of our efforts would be a conditional density function for wheat that is conditioned on height.

The data that we are using comes from the Kellog Biological Station's Long Term Ecological Research Site (LTER). A map of the fields is displayed in Figure 7.1. We periodically took ground truth in eight different fields over the seven days of measurements. These fields were designated fields A through H.

Processing the data as we did before, we subdivided each field into 9 sub–fields, we got 504 70–look data points, for which we had measurements of the crop height. Table 7.1 shows two height measurements from wheat fields A-H taken during the 1995 growing season

We must now decide how many dimensions to represent the densities in. The higher the dimensionality, the better information we have about the different higher-

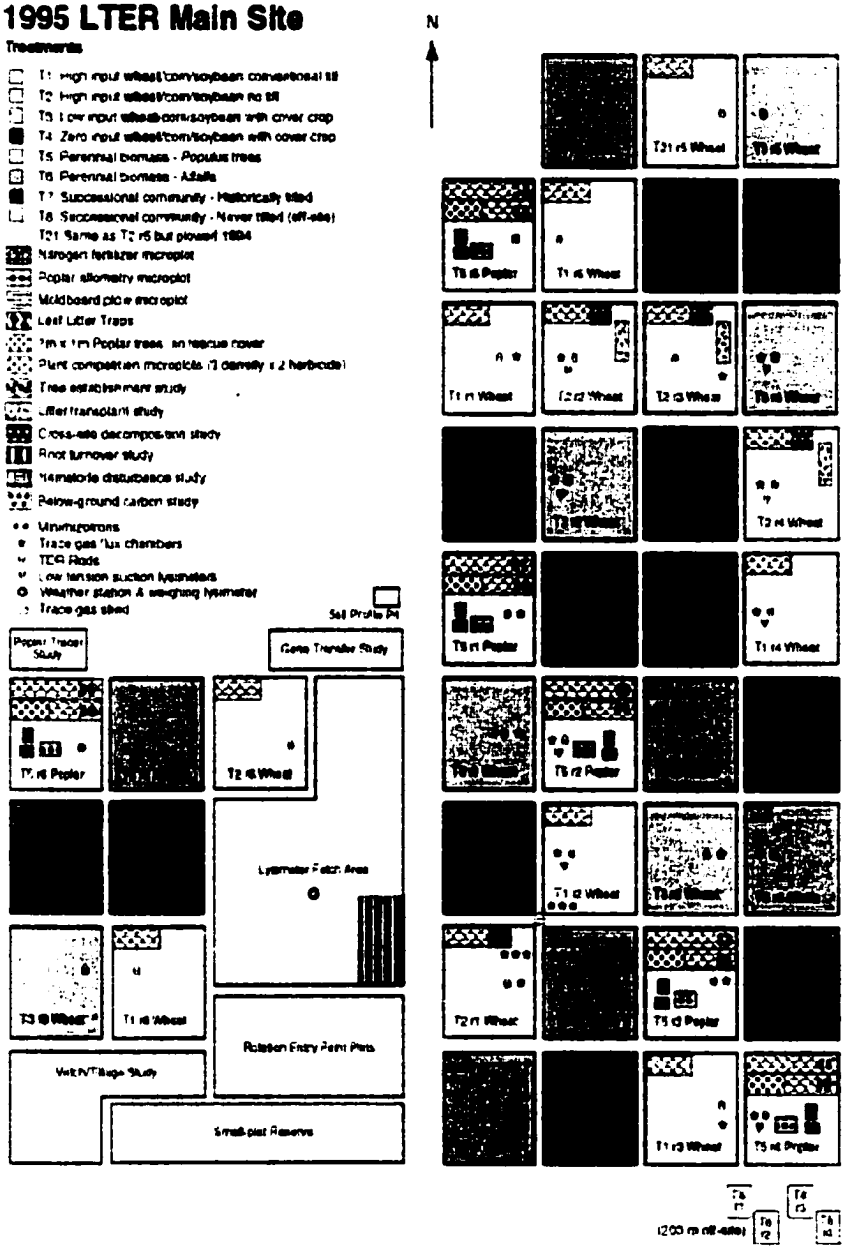


Figure 7.1: Kellog Biological Station Long Term Ecological Research Site. Fields A through H are (respectively): T1R2, T1R4, T2R1, T2R3, T3R3, T3R4, T4R1, T4R2.

order correlations. Figure 7.2 is a plot of a radar measurement versus plant height. It is very difficult, if not impossible to understand what is going on or deal with the parameter estimation problem if we confine ourselves to one dimension. The information that we get from even first order correlations is invaluable to our intuition and understanding in this situation.

The disadvantage of high dimensionality is that we need exponentially more data. For the purposes of this example, and in order to better visualize our data, we choose to represent two dimensions of our measurement. There is no practical reason that we could not represent the wheat in three or four dimensions, if we had more data.

We somewhat arbitrarily choose to plot σ^o_{LHV} and σ^o_{CVV} for the data we have, grouping data points by their height in Figure 7.3.

We took each of these subsets of the data, applied the maximum entropy density estimation technique discussed previously, and plotted the estimated densities in Figures 7.4 through 7.8.

By doing this, we get an intuitive picture of the non-stationarity of the density. The series of pictures we have are snapshots of the statistics at particular heights. Notice the change in direction that the density makes between 49-50 cm, 50-62 cm and 87-91 cm.

The changes in height are not uncorrelated with the other ground truth measurements, such as wet biomass. These correlations may explain the change in direction of movement the density makes as the plant gets taller.

An example of the complicated set of factors underlying the density is how the height is related to water content per unit volume of the plant. Qualitatively, at the

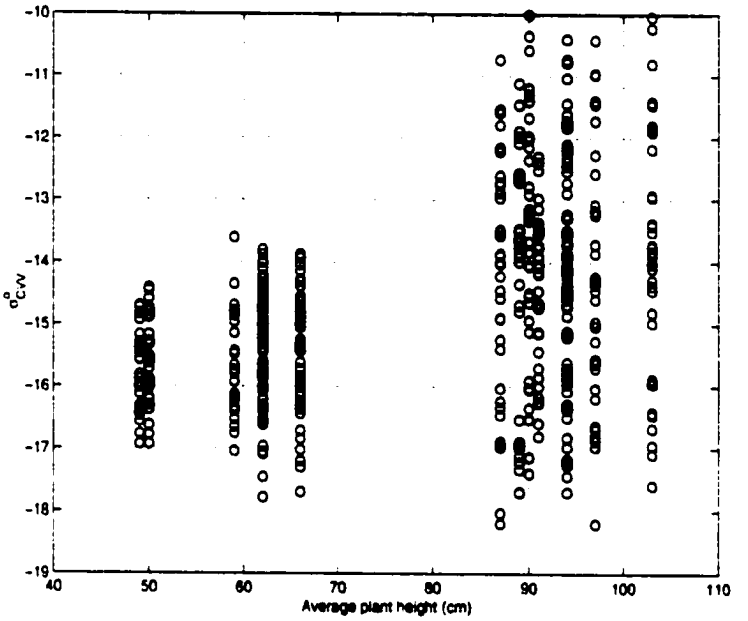


Figure 7.2: Plot of a radar measurement of wheat vegetation versus height.

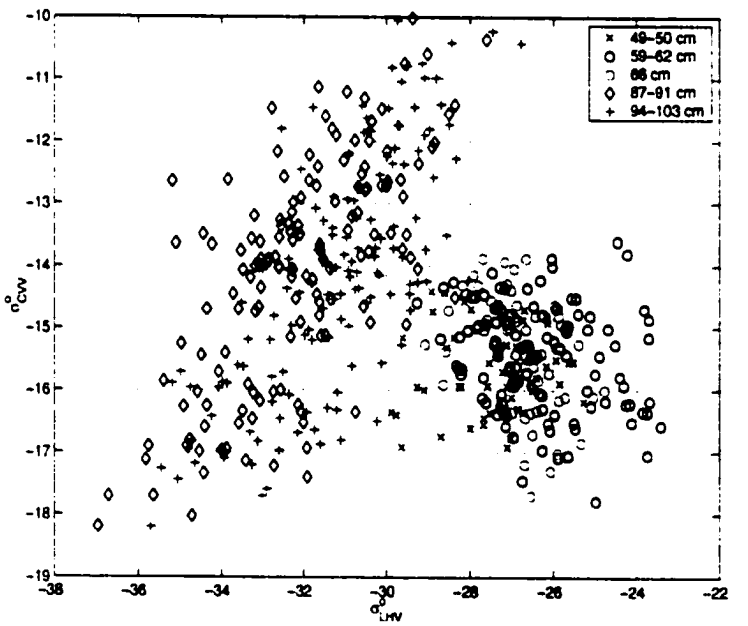


Figure 7.3: Plot radar measurements of wheat vegetation with different heights. The symbol indicates what range the height is in.

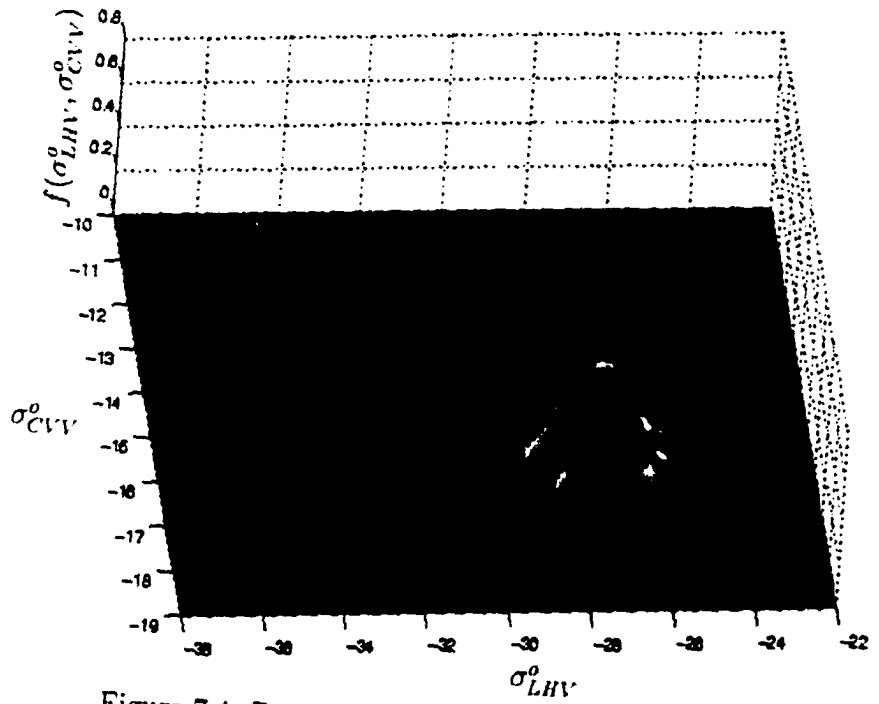


Figure 7.4: Density of wheat approximately 50cm tall.

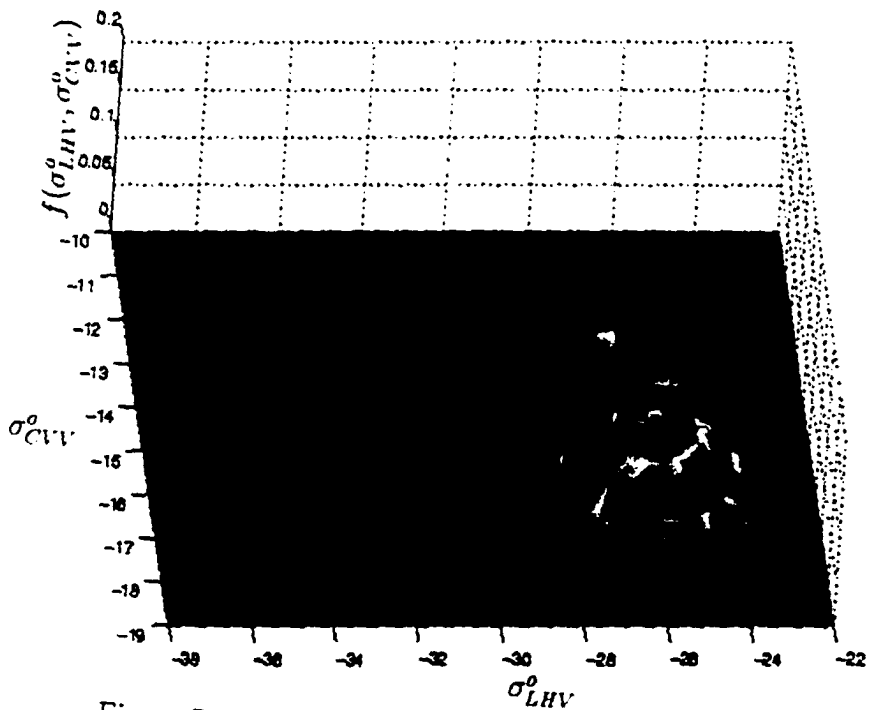


Figure 7.5: Density of wheat approximately 60cm tall.

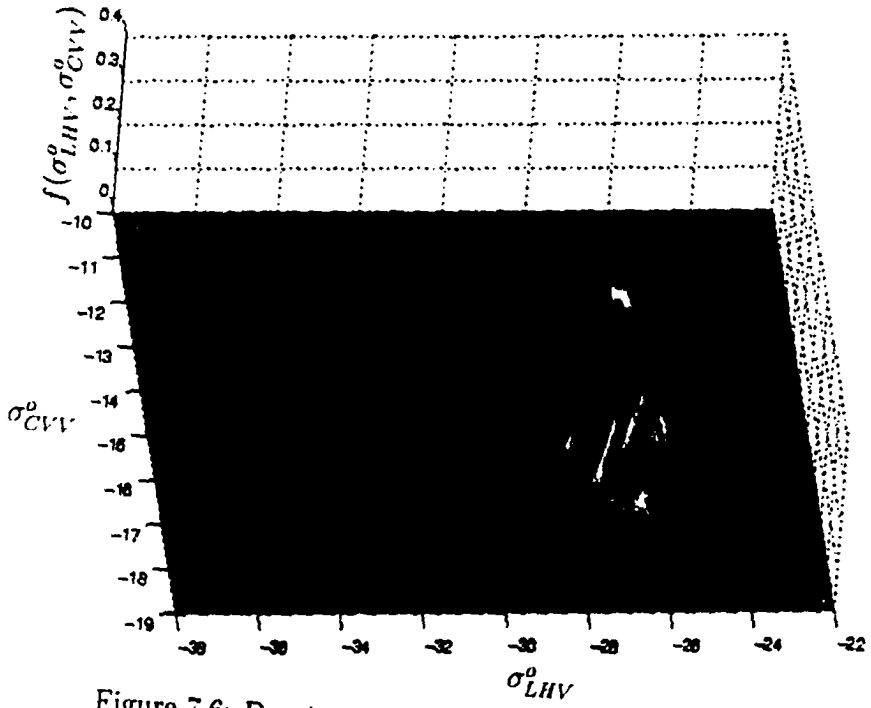


Figure 7.6: Density of wheat approximately 66cm tall.

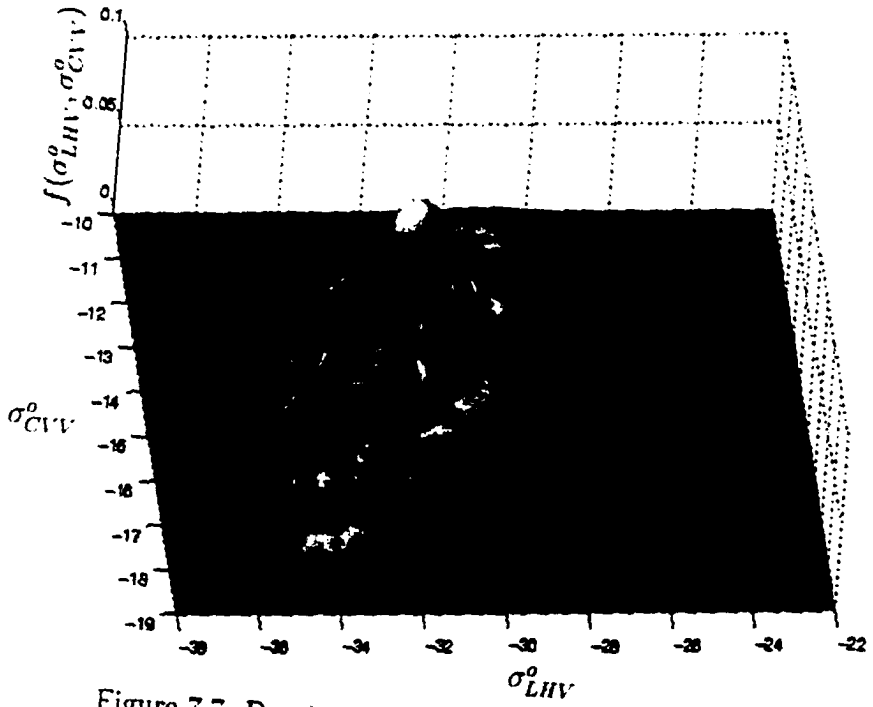


Figure 7.7: Density of wheat approximately 89cm tall.

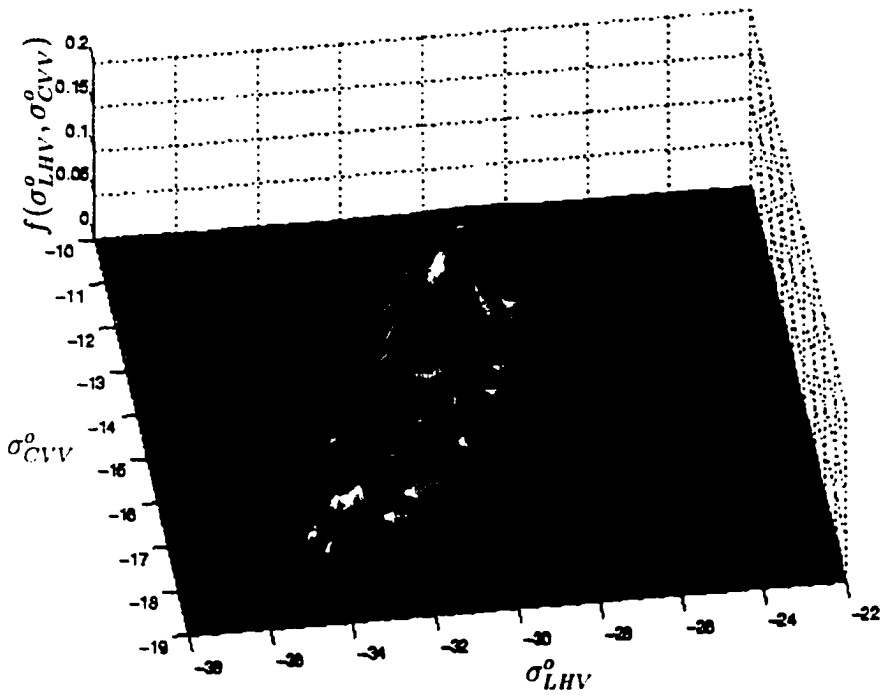


Figure 7.8: Density of wheat approximately 99cm tall.

beginning of its life cycle, the wheat plant is short but green and full of water. The water, with its high dielectric constant, is one of the major factors that alter the response of the radar measurement system. As it proceeds through its growth cycle, the wheat plant gets taller and the water content per unit volume drops, until at its maximum height, it dies and dries out. One would not find a one meter tall wheat plant that has as much water per unit volume as in the beginning of its life cycle.

This relationship is an example of the complicated correlations that go on between the inputs to the system, and can possibly prevent its behavior from being smooth and predictable.

We might note that there is a large gap in the data, between 66 cm and 87 cm. Given a lack of additional data, we can do the linear interpolation we discussed in Equation 7.1 to estimate what the density would look like at 82 cm. In doing this

interpolation, we are assuming that the behavior between 66 and 87 cm is linear, an assumption akin to saying that we have adequately sampled the density in the domain of its parameter. Figure 7.9 shows the two estimated densities with the interpolated density in between them.

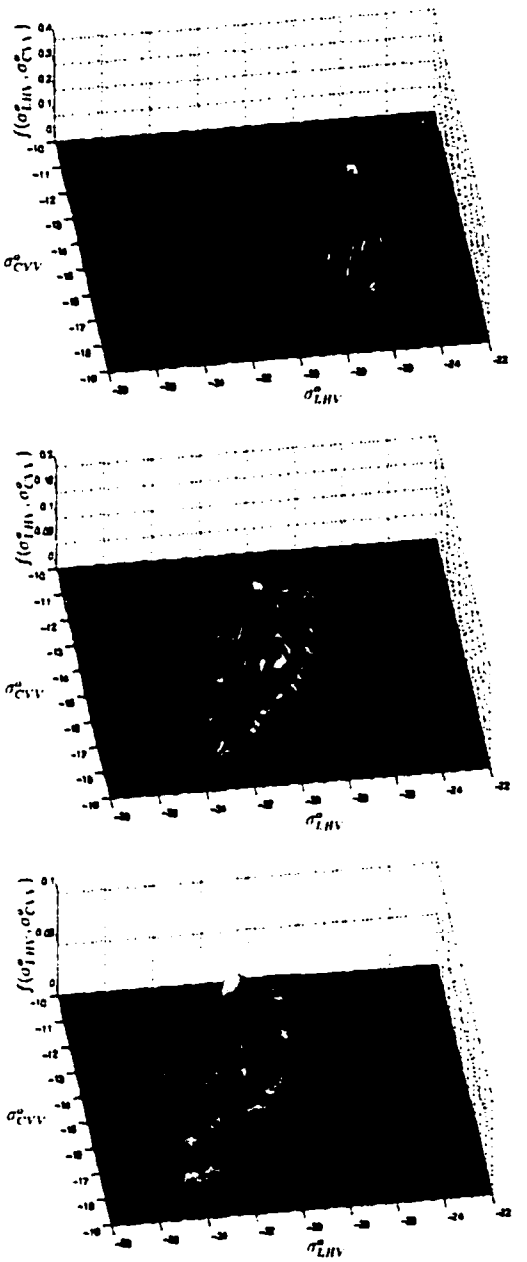


Figure 7.9: Linear interpolation of a probability density. The top figure is the density of wheat approximately 66 cm tall, the bottom figure is the density of wheat approximately 89 cm tall, and the center figure is the interpolated density of wheat approximately 82 cm tall.

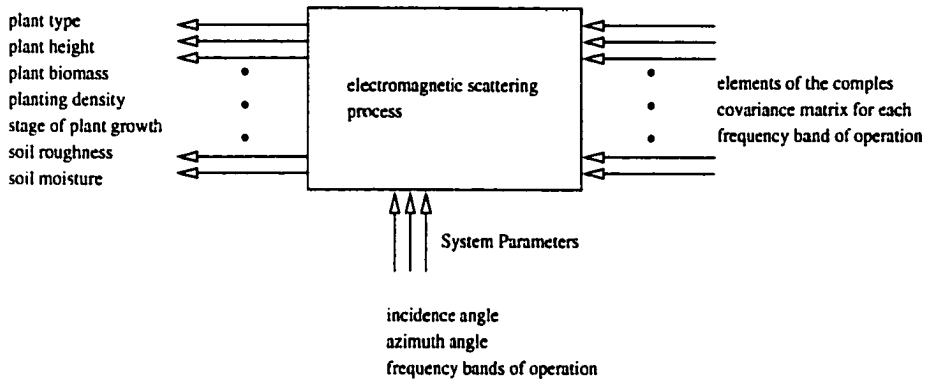


Figure 7.10: Systems level perspective of parameter estimation. Parameter estimation is equivalent to running the electromagnetic scattering process backwards, to find the inputs from the measurements.

	Estimated Height					
True Height	**50 cm**	**60 cm**	**66 cm**	**78 cm**	**89 cm**	**99 cm**
50 cm	39 %	4 %	39 %	19 %	0 %	0 %
60 cm	23 %	31 %	40 %	6 %	0 %	0 %
66 cm	6 %	15 %	67 %	13 %	0 %	0 %
89 cm	0 %	0 %	0 %	8 %	56 %	36%
99 cm	0 %	0 %	0 %	19 %	33 %	48%

Table 7.2: Confusion matrix showing producer accuracies (in percent) for height estimation of wheat using Bayesian classification techniques.

Parameter Estimation

Parameter estimation, in the context of the systems level perspective described when we discussed classification, is the process of running the system backwards, entering the measurements and estimating some of the inputs.

Parameter estimation is a potential application of this parameterized density function, which could give us a rough estimate of the parameter of interest. The mechanism by which we could do this is exactly the same one as we use for classification. Instead of deciding what class we are dealing with, we are deciding what height range we are dealing with.

	Estimated Height					
True Height	**50 cm**	**60 cm**	**66 cm**	**78 cm**	**89 cm**	**99 cm**
50 cm	21	2	21	10	0	0
60 cm	25	33	43	7	0	0
66 cm	3	8	36	7	0	0
89 cm	0	0	0	11	81	52
99 cm	0	0	0	28	47	69

Table 7.3: Confusion matrix showing number of data points in each height bin given by Bayesian classification techniques.

	Height accurately estimated to ±15 cm
50 cm	81%
60 cm	94%
66 cm	100%
89 cm	100%
99 cm	81%

Table 7.4: Table displaying percent of data in each height range classified to within ±15 cm.

In our case we can apply the classification techniques to the parameter estimation problem for the height of wheat, and get the following results. The "classes" in the following confusion matrix are actually height ranges, identified by the range mean, so the "89" class was trained on wheat from 87-91 cm high, while it represents wheat from 84-94 cm. Also note that the 78 cm category was interpolated, as discussed above, because we had no data to characterize it with.

To evaluate the feasability of our idea, we ran a standard Bayesian classifier with the two dimensional densities developed for each height range. The results are summarized in Tables 7.2 and 7.3.

Table 7.4 condenses these confusion matrices into a table which evaluates the accuracy of our parameter estimation. We tally the percent of data that we estimated to within ±15 cm of the actual height.

Considering the small amount of data we trained with and the few dimensions that we used, this parameter estimation gives us relatively good results.

7.3 Point Target Detection

The second of the tasks that we wish to consider is the detection of a point target in a vegetated background. Unlike a distributed target, which typically covers an area of the ground, a point target is one that is localized, such as a truck or the jutting peak of a craggy cliff. Because of their different nature, point targets do not have the same fading statistics as distributed targets do.

It can be very difficult to distinguish point target pixels from non-point target pixels. The problem for this application is to correctly identify known point targets while minimizing the chances of accidentally detecting a point target where there is none.

7.3.1 Using Multiple Correlations by Adding Dimensions

The simplest improvement we can offer the point target detection process is the application of the maximum entropy density estimation technique to finding the probability density function of the vegetation.

Figure 7.11 is a plot of a point target density superimposed on a vegetation pixel density (we have chosen our measurement to be σ^o_{LHH}). To distinguish a point target versus a non-point target based on this information, we would locate a measurement on the x-axis and choose the class (either point target or vegetation) that is most

probable, or has a higher value at this value. This can be interpreted as a Maximum likelihood technique, or a two class optimal Bayesian classification with uniform prior probabilities. This can also be seen as a thresholding technique.

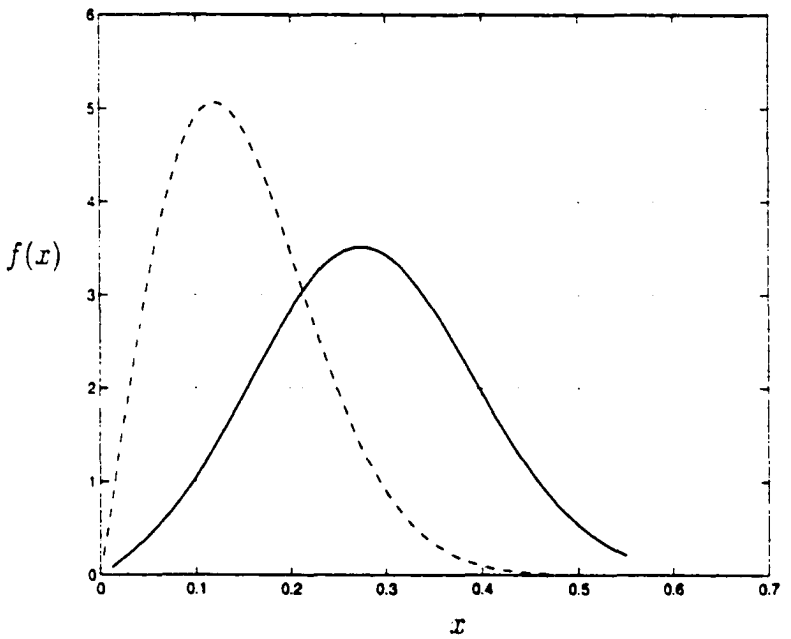


Figure 7.11: Point target discrimintation. This is a plot of the simulated density of radar measurements of vegetation (dashed curve) superimposed with the density of radar measurements of a point target (solid curve). The vegetation mean is at -18 dB and the point target mean is at -16 dB.

Using the density estimation technique developed above offers a computationally efficient way of

1. The natural spatial and temporal variability of the vegetation measurements

2. The higher dimensional correlations in the vegetation pdf

These items are often neglected when doing point target detection. When the spatial and temporal variability are taken into account, as in [22], the methods used are typically parametric, as opposed to the non-parametric density estimation we propose.

One Dimension

	Classified As	
True Class	Point	Veg
Point	72%	28%
Veg	21%	79%

Two Dimensions

	Classified As	
True Class	Point	Veg
Point	82%	18%
Veg	14%	86%

Table 7.5: Theoretical accuracy in distinguishing point targets versus vegetation pixels, with one and two dimensional densities.

Table 7.5 shows the increase in expected accuracy that we gain by making our densities two-dimensional, for the hypothetical example offered in Figure 7.11. (The pdfs of the second dimension have the same means as those of the first.) The increase in accuracy is due to the inclusion of higher order correlations in the point target detection thresholds, which can now be applied in two dimensions.

Figure 7.12 is a plot of a density estimated from real alfalfa data. Note that unlike our idealized example, this density includes the temporal and spatial variation in the vegetation measurements. This type of representation is a significant improvement over the simplified model shown in Figure 7.11.

7.3.2 Using Information About the Non-stationarity of Vegetation for better modeling of Point Targets in a Vegetation Background

In this section, we present the mechanics of more accurately modeling what we would measure if we have a point target against a vegetation background. To do this, we will include the effect of non-stationarity in the vegetation measurements in the point target pixels. We present these techniques for a sensor that is not polarimetric

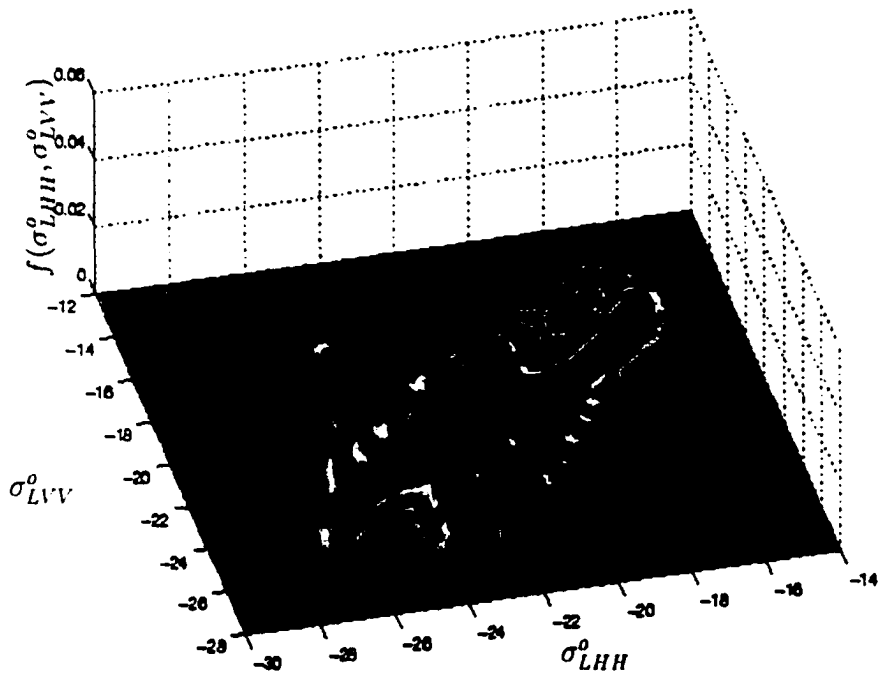


Figure 7.12: Density of radar measurements of alfalfa. The axes are σ^o_{LHH} radar measurements versus σ^o_{LVV} measurements.

or multifrequency and only has a single channel.

Consider an image, and call our one dimensional measurement M. This can actually be σ^o_{LHH} or any other magnitude channel that we have available to use.

We will make the following simplifying assumptions:

1. Each pixel of vegetation (also called clutter in this context) in a SAR image contains a large number of randomly distributed scatterers, no one of which dominates in strength.

   When we look at the envelope of vegetation measurements using linear detection (as we will in this section), we find that our measurements with fixed biophysical parameters have a Rayleigh distribution [38]. Thus, the probability density

function for a one-look measurement of vegetation is,

$$K_{(1)}(M = m|\widehat{M} = \mu) = \frac{\pi m}{2\mu^2} e^{\frac{-\pi m^2}{4\mu^2}} \tag{7.2}$$

where the subscript (1) denotes that it is a one look measurment, and m is what our sensor measures. The parameter μ is what the sensor would measure in the absence of speckle.

2. Each pixel which contains a point target in a SAR image contains a large number of randomly distributed scatterers (produced by the vegetation background) with one or more strong scatterers that are fixed.

   We rationalize the deterministic value of the strong point target scatterers by assuming that our point targets have a particular, fixed orientation with respect to the radar measurement system.

   In this application, we will consider a point target with a single, fixed scatterer. The probability density function of this scatterer is given by the Nakagami-Rice density function, as given in[38], Volume 2 on page 483,[3]

$$L_{(1)}(M = m|\widehat{M} = \mu) = \frac{\pi m}{2\mu^2} e^{\frac{-\pi(m^2+A^2)}{4\mu^2}} I_0\left(\frac{\pi m A}{2\mu^2}\right) \tag{7.3}$$

   where I_0 is the modified Bessel function of the first kind with order zero. As before, m is what we measure, and μ is what we would measure if there were no fading statistics and we were measuring vegetation. This density represents the fact that the electromagnetic measurement is simply the sum of the return from

[3]More complicated densities that reflect our measurements of point targets with multiple scatterers can also be computed and used where we are using $L(\cdot)$.

a regular pixel added to the deterministic, fixed return from our point target scatterer.

To predict the probability density function of a point target in a particular background, we will use the following equations:

$$r_{(n)}(M = m) = \int K_{(n)}(M = m|\widehat{M} = \mu)t(\mu)d\mu \tag{7.4}$$

$$v_{(n)}(M = m) = \int L_{(n)}(M = m|\widehat{M} = \mu)t(\mu)d\mu \tag{7.5}$$

where n is the number of looks in our data, and

- $K(M = m|\widehat{M} = \mu)$ and $L(M = m|\widehat{M} = \mu)$ are the known pdfs that represent the statistics of a single vegetation pixel and a single point target pixel, respectively, with a fixed, nonstationary value. The uncertainties in these pdfs are due only to fading effects.

- $r(M = m)$ and $v(M = m)$ are the pdfs of measurements that we would record, out in the field, for clutter pixels and point target pixels, respectively.

- $t(\mu)$ is the probability density function which represents the uncertainty due to biophysical parameter variation in the clutter pixels.

The integrals that we have written are both a standard method of removing the conditioning on a probability density function.

They are also conveniently in the form of a convolution. This gives us the intuition that our final density is the superposition of many individual densities (given by our kernels $K(\cdot)$ and $L(\cdot)$) each with some weighting factor.

Using the density estimation techniques of Chapter 5, we can take a set of clutter data and estimate $r(M = m)$. Both $K(M = m|\widehat{M} = \mu)$ and $L(M = m|\widehat{M} = \mu)$ are known pdfs.

We would like to use Equation 7.4 to solve for $t(\mu)$, and then substitute $t(\mu)$ in Equation 7.5 to find $v(M = m)$. $v(M = m)$ is the pdf describing the statistics of a measurement of a pixel containing a point target.

One could also use this technique to automatically develop point target recognition algorithms for finding the point target in a background that was uncharacterized at the time of system deployment.

Deconvolution Theory

The astute reader will note that finding $t(\mu)$ from Equation 7.4 is not a trivial task. If the integral is a convolution, finding $t(\mu)$ is a deconvolution.

Solving for $t(\mu)$ constitutes solving a Fredholm integral equation of the second kind. These integral equations are notoriously ill conditioned, which means that it is unlikely that we will find a unique solution. Figure 7.13[4] is an example where deconvolution could fail with a kernel that interests us. As the figure illustrates, multiple unique solutions exist, and a deconvolution technique could give us either of the perfectly valid results pictured, which despite solving the integral, are totally

[4]The kernels in Figure 7.13 are all produced assuming that the data is taken with square law detector. In that case, the underlying Gaussian distribution of the scattering matrix elements becomes exponential. For the purposes of deconvolution, it is more convenient to represent the data as if it was taken with a linear detector, since the standard deviation of the resulting density goes down quicker with increased number of looks. With the linear detection scheme, the underlying Gaussian distribution becomes Rayleigh, and we will assume that our data is linearly detected for the remainder of the discussion.

different from one another.

Tremendous uncertainty in the single look distribution (reflected by a large standard deviation) effectively prevents us from applying deconvolution techniques to data which has too few independent samples. Multiple looks narrow the kernel of the integral equation and reduce the possible solutions, increasing our chances of finding a solution that is close to the the actual solution.

To do the deconvolution, we will apply the principles of maximum entropy to find the most likely density function from the family of possiblities, and get our fading-free representation. The idea is to minimize $|r(M = m) - \int K(M = m|\widehat{M} = \mu)t(\mu)d\mu|^2$ while simultaneously minimizing the entropy, $-H(t(\mu))$. This gives us the maximum entropy solution, which given no extra information, is a statistically sound guess. It is important to note that we are assuming something about the smoothness and character of the vegetation texture in doing this type of deconvolution.

To perform this maximization, we cast our equations into matrix form. We will cast the kernel into a matrix that we will call K, the measured density will be the vector r and the unknown density which we wish to deconvolve will be t.

We will call

$$V = |r - Kt|^2 \tag{7.6}$$

$$W = \sum_i t(i) \log t(i) \tag{7.7}$$

Now we must find a t which minimizes

$$V + \lambda W \tag{7.8}$$

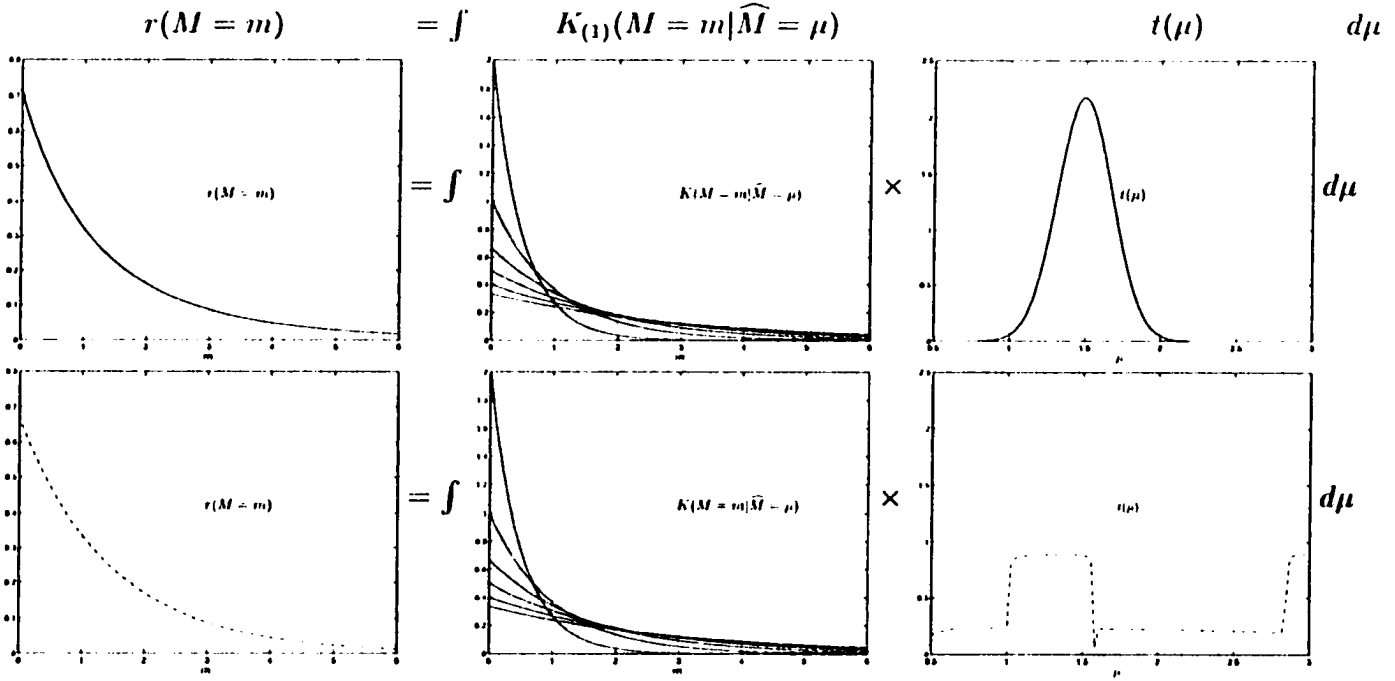


Figure 7.13: Each line is a graphical depiction of the functions in the deconvolution integral. As the reader can see, deconvolution doesn't always work. Note the two different solutions, $t(\mu)$, essentially produce the same result, $r(M = m)$.

The reader will recognize that this minimization is simply a nonlinear application of the general regularization principle, which is commonly used in inverse problems.

The nonlinearity of the second term is a significant problem, which is solved by a technique proposed by Skilling in [35] (and discussed briefly in Appendix B) for the purpose of image enhancement.

Deconvolution Example

In this section we will use deconvolution to implement the ideas discussed in Section 7.3.2, to find the statistics due to clutter non-stationarity, and use it to predict the behavior of a point target with that clutter background.

Assume that we have a set of representative background clutter data. We also know the amplitude of the point target we would like to detect, which is a single dominant scatterer.

To do point target detection, we choose an image the image with a low number of independent samples, but high enough to do deconvolution. Assume that our image starts out as a five look image (n=5). After running our maximum entropy density estimation technique on the vegetation data, we generate a density $r_{(5)}(M = m)$. Combined with our knowledge of $K_{(5)}(M = m|\widehat{M} = \mu)$ we performed the deconvolution of $t(\mu)$, following the algorithm in Figure 7.14, and got the results shown in Figures 7.15 and 7.16.

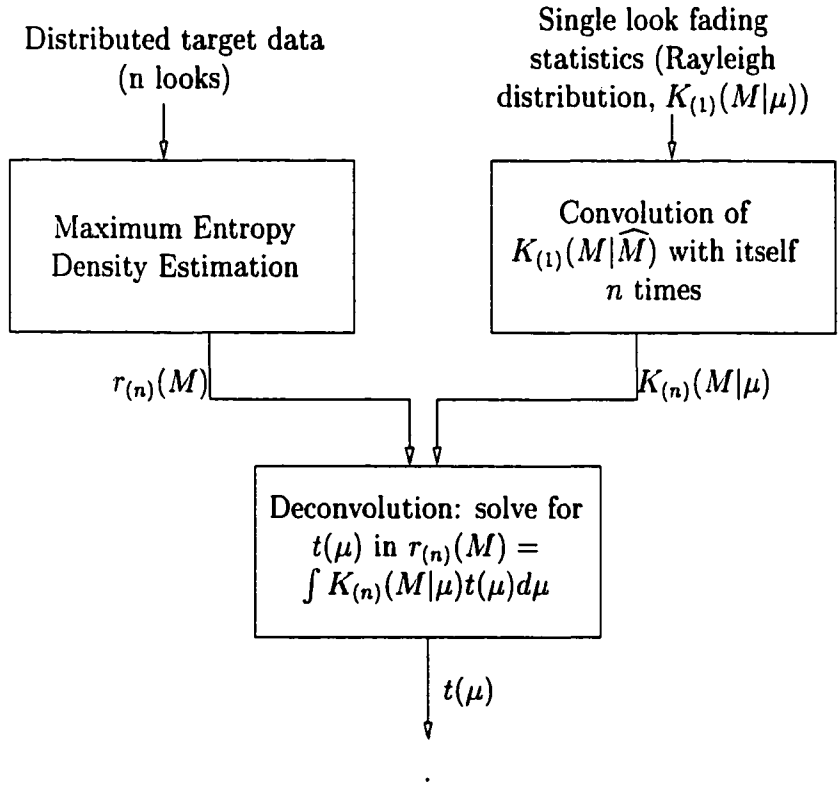


Figure 7.14: A flowchart for solving for $t(\mu)$ from a single, characteristic image by deconvolution.

Now that we have estimated $t(\mu)$, we can simply use in in Equation 7.5 to find the pdf that we expect from the point target in this background.

Equations 7.4 and 7.5 were used to generate the plots in Figure 7.17. These pdfs include the clutter background statistics.

Figure 7.18 shows the same pdfs if we only use the clutter mean and do not take the non-stationarity of the clutter into account. The pdfs are considerably different, and less accurate.

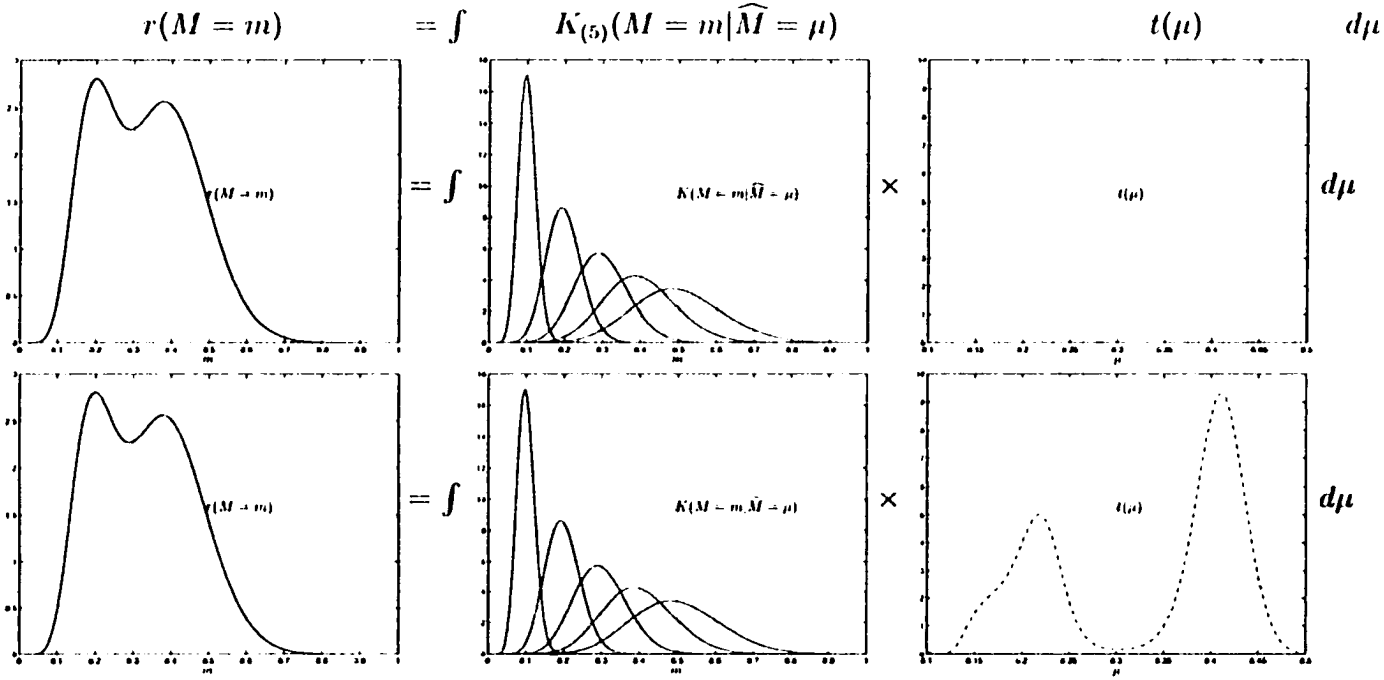


Figure 7.15: Deconvolution in a realistic situation. The top row is a description of what we begin with (we need to find the function in the integrand on the right, and the bottom situtation is after deconvolution. The result of the deconvolved function being integrated is compared to the original curve. Note that they overlay almost completely.

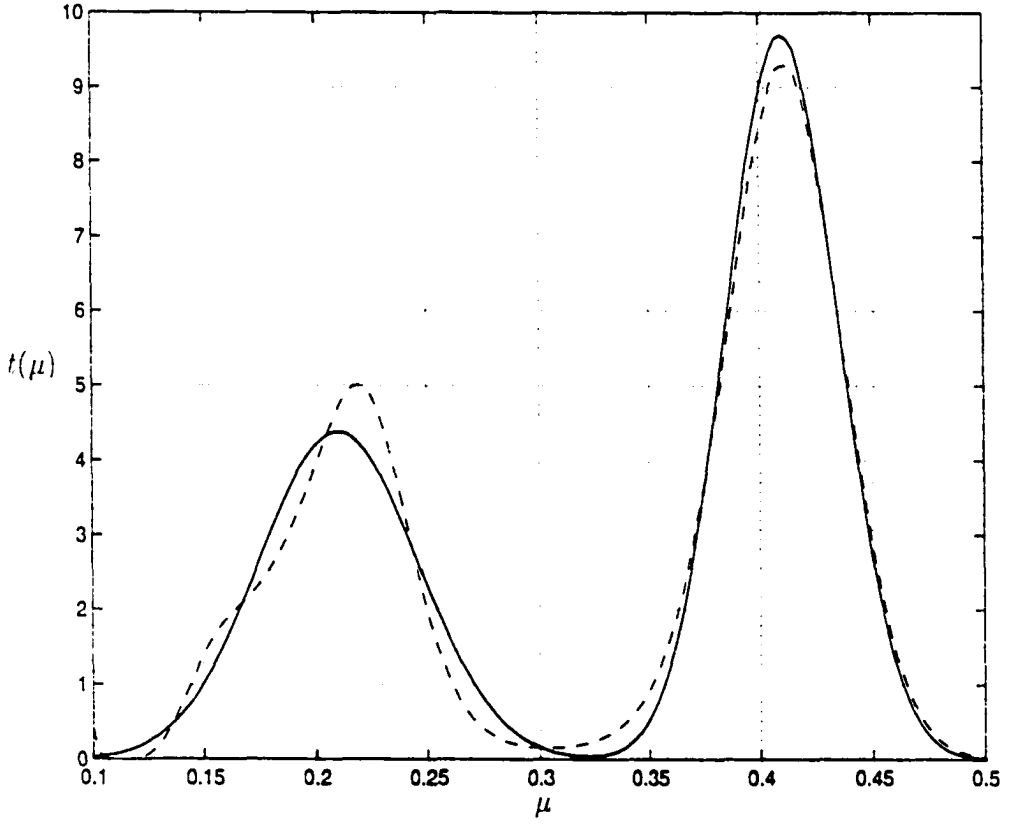


Figure 7.16: Plot of the $t(\mu)$. This is the texture density that describes the non-stationarity of the simulated vegetation. This is a comparison of the actual density (solid line) versus the deconvolved estimate of the density (dashed line).

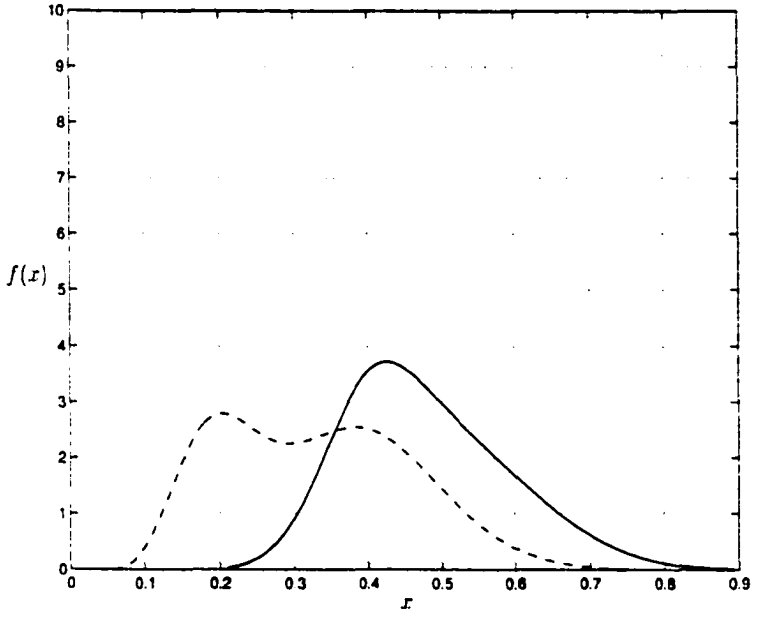


Figure 7.17: The density of a vegetation background (dashed line) pixel is plotted with the density of a point target pixel (solid line.) Texture statistics of the vegetation are included in both the vegetation density and the point target density. Both densities are for a five look image, where the five independant looks were provided by the sensor bandwidth.

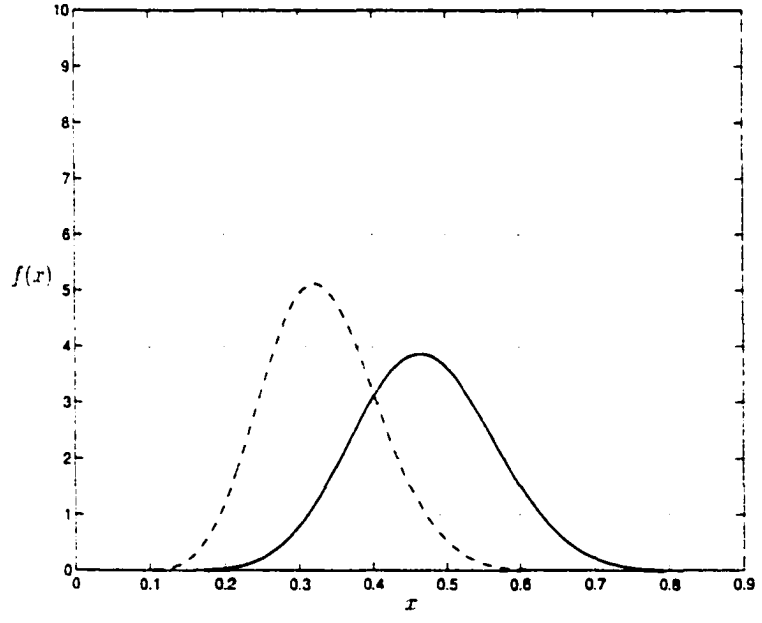


Figure 7.18: The density of a vegetation background (dashed line) pixel is plotted with the density of a point target pixel (solid line.) The vegetation is assumed to be homogeneous, with no texture. Both densities are for a five look image, where the five independant looks were provided by the sensor bandwidth.

CHAPTER 8

Conclusions and Future Work

We have developed a technique called the Hierarchical–Bayesian classifier, which we demonstrated has the ability to classify short vegetation using multi-channel SAR data to an overall accuracy of 93%. The most outstanding feature of the classifier is the fact that the same classifier can be used throughout the growing season with no modifications or retraining.

We have developed a density estimation technique based on maximum entropy methods that is, for most practical cases (and certainly for our examples), more accurate, more computationally efficient, and more efficient in terms of storage than the kernel density estimator.

We have demonstrated both a technique of representing the non–stationarity of complicated targets, as well as a deconvolution technique which could be applied to remove speckle from the statistical representation of non–stationary targets. These techniques could be applied to parameter estimation and point target representation, respectively.

There are various directions one could go in to extend the work described here:

1. The assumptions of any methodology require careful scrutiny, and the techniques presented here are no exception. Both the classification technique and the parameter estimation technique assume that the data used to describe the classes we are distinguishing between spans the domain of all the conditions that it could possibly have.

   Scrutiny of the data sets as well as ground truth to ensure this complete representation are necessary for robustness of this technique.

2. Furthermore, the limits of the deconvolution technique need to be explored. There is no guaranteeing a correct solution for deconvolution techniques, and certain kernels will cause deconvolution to fail. (Notably, one-look kernels will sometimes create solutions that satisfy everything, but despite all of our efforts, are wrong.)

   More study needs to be done to establish how much confidence we can have in each deconvolution we do.

3. The classification technique, which we have so successfully applied to short vegetation, could easily be applied to other distributed targets and remote sensing data. In addition, the generality and statistical foundation upon which the technique rests allows for potential applications in many other fields of science and engineering requiring classification.

4. The density estimation technique is also a very general technique, which, like the classification technique, is not inherently wedded to remote sensing problems. The beauty of generality is its ability to be reused in situations that were not originally envisioned.

   The properties of the maximum entropy density estimation technique presented here might be useful in many other problems, and for many other applications.

APPENDICES

APPENDIX A

What is Entropy?

A natural question one might ask is "What is entropy, anyway?" The main difficulty with the concept of entropy is that the word has been used by different people to mean apparently different things. There are four people that I will concentrate on who have developed definitions of the word 'entropy' in a scientific context. They are R. J. E. Clausius, L. Boltzmann, J. W. Gibbs, and C. E. Shannon. The former three were working in the field of thermodynamics – the study of heat and mass transfer. The latter was creating the field of information theory, which at first glance, appears unrelated to the others.

A.1 Thermodynamic Entropy

In 1850, Clausius defined a property of a thermodynamic system that he called 'entropy'. We will call this Clausius' entropy. This definition is an experimental one, stating that the change in entropy is the change in heat in a system divided by absolute temperature. Specifically, a change in entropy, H is given by $dH = dQ/T$, where Q is heat, and T is absolute temperature.

Boltzmann approached thermodynamic analysis by attempting to develop a microscopic model that agreed with existing macroscopic models. He quantified the position and velocity of each molecule and applied statistical techniques, hoping to find classical thermodynamics relations. The equation that he identified as entropy only agrees with Clausius' entropy when the system is a volume of an ideal gas.[1]

Gibbs, working independently from Boltzmann, also attempted to model ther-

[1] [11]

modynamics from a microscopic perspective, (quantifying the position and velocity of each molecule) but came up with a different answer. The statistical mechanics developed by Gibbs accurately represents the experimental concept of entropy that Clausius defined. When maximized, Gibbs' entropy is always equal to Clausius' entropy.

The statistical mechanics point of view of entropy is commonly referred to as the amount of disorder in the system. In this interpretation, the disorder of a system depends on the state of the system at the time of measurement. The disorder is proportional to how many ways a system can realize that particular state, and this is what entropy is measuring.

For example, consider a system whose state is characterized by adding up the number of heads in five fair coin tosses. We will call this sum the macrostate of the system, while we will call the specific sequence of coin tosses (using T for tails and H for heads – e.g. THHTH) the microstate. The entropy of a system is proportional to how many microstates realize the particular macrostate that the system is in.

In this system, the 0 macrostate can only be achieved by one microstate { TTTTT }, and thus this is a very ordered state of low entropy. On the other hand, the 2 state can be realized by ten microstates, { HHTTT, THHTT, TTHHT, TTTHH, HTHTT, THTHT, ... }, and thus is a more disordered state of high entropy.

Applying this concept to a physical system (such as a volume of diffuse gas) gives us a conceptual picture of what thermodynamic entropy means, and also explains why you can start a physical system in a state of low entropy, and it will tend inevitably towards higher entropy states. Statistically speaking, with the number of particles

that we are dealing with in a gas, the system will tend towards states of high entropy.

A.2 Information Theoretic Entropy

Shannon is the last person on the list to define the word entropy. While writing "A Mathematical Theory of Communication," he defined the entropy of a discrete[2] random variable X as

$$H(X) = -\sum_a p_a \log_2 p_a \tag{A.1}$$

where p_a is the probability that the outcome of the random variable is symbol 'a'. He indicates that entropy is a function of a random variable, but it would also be accurate to write $H(\mathbf{p})$, where $\mathbf{p}$ is a vector representing a discrete probability density function for that random variable.

In the context of what Shannon was doing, this entropy is the expected value of information conveyed by a random variable, and is exactly the same mathematical expression as the formulation of Gibbs' entropy in statistical mechanics. By reading Shannon's thesis, it appears that he named entropy solely on the basis of this mathematical similarity,[3] without understanding the deeper connection.

[2]Here we will develop entropy for the probability density function of a discrete random variable, but it can easily be extended to work with continuous random variables

[3]Myron Tribus wrote that Shannon had "made it quite clear to me that he considered applications of his work to problems outside of communication theory to be suspect and did not attach fundamental significance to them."

Myron Tribus also writes about

> ... having asked Dr. Shannon what his reaction had been when he realized that he had identified a measure of uncertainty. Shannon said that he had been puzzled and wondered what to call his function. ... Shannon said he sought the advice of John von Neumann, whose response was direct, "You should call it 'entropy' for two reasons: first, the function is already in use in thermodynamics under the name; second, and more importantly, most people don't know what entropy really is, and if you use the

For the reader not familiar with the concept of the information transmitted by a random variable, a quick explanation follows. Random variables, in and of themselves, do not contain information. Observing a series of random numbers from a unit variance Gaussian random variable, (usually written as $N(0,1)$) does not tell us anything useful. It is only in the context of a communications system that people can talk about the information conveyed by a random variable.

Imagine a communications system with one source and one receiver. The source sends symbols to the receiver, with each symbol having a particular meaning, as well as a particular frequency of occurrence. To the source, the symbols are chosen to convey particular meanings, so they are chosen deliberately, but to the receiver, who does not know what the source is about to send, each symbol is best viewed as the outcome or realization of a random variable.

Call the random variable that conveys the symbols, X, and the frequency of the symbol 'a' occurring as p_a. The information is given by $I(X = a) = -\log_2 p_a$. This description is a monotonic function of probability, such that the information measure increases as the probability of the symbol decreases. The rarer or more surprising the symbol is, the more information you get from it. There are many monotonic functions that satisfy this requirement; i.e. it is also possible to define information more simply as $1/p_i$. However, the logarithm is more mathematically suitable and easier to maximize.

Shannon's measure of information is, at a very basic level, a measure of uncer-

word 'entropy' in an argument you will win any time!"

Both quotes are from [37], p 1,2.

tainty. To illustrate, consider the experiment where a source in Perth sends a symbol to a receiver in Ann Arbor. It has been previously agreed that symbol 'a' means "the sun has risen today" where symbol 'b' means "the sun has failed to rise today." Receiving symbol 'a' conveys almost no information, since the receiver can predict with high certainty that it will be sent, even before it is sent. Symbol 'b,' on the other hand, conveys a lot of information; something unexpected has happened in Australia or possibly even in our solar system.

In the aftermath of Shannon's thesis, E. T. Jaynes wrote a series of articles, which include [14, 15] that presented entropy as an abstract statistical concept.

A.3 Similarities Between the Different Entropies

Shannon's measure of uncertainty, as well as Gibbs' measure of the disorder seem somewhat unconnected, but both give a glimpse of the underlying statistical concept that Jaynes pointed out. From a purely logical point of view, just because two concepts give rise to the same mathematical formula, this does not indicate that the concepts are the same.

In this case, however, there is a common underlying abstract idea, which ties together information theoretic entropy and thermodynamic entropy. We will demonstrate the equality of the two concepts by using showing their interchangability, that each can be substituted for the other. We will do this by using each in the context that the other was made in.

Lets look at the information theory perspective in terms of microstates. Assume

we have a communication system with a series of symbols. Instead of assigning our standard messages to those symbols, we are going to transmit them in bursts, and each burst will contain the location and velocity of N particles of gas in a closed container. Now, each burst is equivalent to a microstate of the physical system. We know that the physical system will naturally go towards the high entropy states, and if we measure the Shannon entropy of each burst over time, it will represent the disorder of our physical system, just like Gibbs entropy.

On the other hand, we can look at a physical system in terms of the information content it conveys for a particular entropy. Take a volume of gas, (a cubical container) and with it, we create a hypothetical communication system with it. Assume we can somehow freeze the Gibbs entropy of the system. Now, for a particular value of Gibbs entropy, there are a fixed number of micro states that we can choose from. If we assign a meaning to each microstate, then the higher the Gibbs entropy the more information we can convey, because we have more symbols at our disposal. Putting the gas in a low entropy state limits the number of configurations that can give rise to that state, and limits our ability to transmit information with it. Now, the measure of Gibbs entropy in the system is behaving exactly like Shannon entropy for a communications system.

The underlying idea that is the same in each case is that there is a version of entropy that is a measure of probability density functions. It measures the uncertainty or disorder of number of ways that we can get to this particular macro state. This statistical version of entropy established a conceptual connection between Shannon's entropy and thermodynamic (Gibbs' and Clausius') entropy. Using this concept and

the maximum entropy principle that Jaynes put forth, Gibbs entropy and Shannon entropy can be viewed as a special case of statistical entropy. Now, the statistical concept of entropy can usefully be applied to thermodynamics, information theory, and image processing.

APPENDIX B

Skilling's Nonlinear General Regularization

The following formulation of Skillings nonlinear regularization closely follows an explanation in Numerical Recipes [26]. We begin with the problem of trying to minimize

$$V + \lambda W \tag{B.1}$$

where

$$V = |r - Kt|^2 \tag{B.2}$$

$$W = \sum_i t(i) \log t(i) \tag{B.3}$$

This technique starts with a discretized version of our solution vector, t. Each element of this vector can be thought of as a dimension in the solution space of the deconvolution problem. Skilling, *et al.* choose a three dimensional subspace defined by vectors **a**, **b** and **c**, whos elements are described by the following equations:

$$a(i) = t(i)\frac{\partial V}{\partial t(i)} \tag{B.4}$$

$$b(i) = t(i)\frac{\partial W}{\partial t(i)} \tag{B.5}$$

$$c(i) = \frac{t(i)\sum_j \left(\frac{\partial^2 V}{\partial t(i)\partial t(j)}\right) t(j) \left(\frac{\partial W}{\partial t(j)}\right)}{\sqrt{\sum_j t(j) \left(\frac{\partial W}{\partial t(j)}\right)^2}} - \tag{B.6}$$

$$\frac{t(i)\sum_j \left(\frac{\partial^2 V}{\partial t(i)\partial t(j)}\right) t(j) \left(\frac{\partial V}{\partial t(j)}\right)}{\sqrt{\sum_j t(j) \left(\frac{\partial V}{\partial t(j)}\right)^2}} \tag{B.7}$$

where the subscript i indicates which element of the vector we are referring to. A standard minimization technique is performed in this subspace (we used a downhill

simplex minimization) and a new subspace is computed based on the current minimum.

This procedure converges, having simultaneously minimized the error and maximized entropy, and we have extracted the texture density from our raw vegetation density.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] "AIRSAR integrated processor documentation: Data formats,", April 1995. Version 0.01.

[2] G. H. Ball and D. J. Hall, "Isodata, a novel method of data analysis and pattern classification," Technical report, Stanford Research Institute, 1965.

[3] J. M. Borwein and W. Huang, "A fast heuristic method for polynomial moment problems with boltzmann-shannon entropy," *SIAM Journal of Optimization*, vol. 5, no. 1, pp. 68-99, 1995.

[4] J. M. Borwein, A. Lewis, and D. Noll, "Maximum entropy reconstruction using derivative information, part 1: Fisher information and convex duality," *Mathematics of Operations Research*, vol. 21, no. 2, , 1996.

[5] L. Chao, "Multidimensional nonstationary maximum entropy spectral analysis by using neural net," *Mathematical Geology*, vol. 31, no. 6, , 1999.

[6] S. R. Cloude and E. Pottier, "An entropy based classification scheme for land applications of polarimetric SAR," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 35, no. 1, , January 1997.

[7] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley, 1991.

[8] R. de Bruin, D. Salomé, and W. Schaafsma, "A semi-Bayesian method for non-parametric density estimation," *Computational Statistics and Data Analysis*, vol. 30, pp. 19-30, 1999.

[9] L. Devroye, *A Course in Density Estimation*, Birkhäuser, 1987.

[10] C. Fraley and A. E. Raftery, "How many clusters? which clustering method? answers via model-based cluster analysis," Technical Report 329, University of Washington, Department of Statistics, Box 354322, Seattle, WA 98195-4322, U.S.A., February 1998.

[11] S. F. Gull, "Some misconceptions about entropy," in *Maximum Entropy in Action*, Clarendon Press, 1991.

[12] P. Hall and B. Presnell, "Density estimation under constraints," *American Statistical Association*, vol. 8, no. 2, pp. 259–277, 1999.

[13] Y. Ito and S. Omatu, "Polarimetric SAR data classification using competitive neural networks," *International Journal of Remote Sensing*, vol. 19, no. 14, pp. 2665–2684, 1998.

[14] E. T. Jaynes, "Information theory and statistical mechanics," *Physical Review*, vol. 106, pp. 620–630, 1957.

[15] E. T. Jaynes, "On the rationale of maximum-entropy methods," *Proceedings of the IEEE*, vol. 70, pp. 939–952, 1982.

[16] J. R. Jensen, *Introductory Digital Image Processing*, Prentice Hall, second edition, 1996.

[17] B. Jeon and D. A. Landgrebe, "Partially supervised classificatoin using weighted unsupervised clustering," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 37, no. 2, , March 1999.

[18] Y. Kouskoulas, L. Pierce, F. Ulaby, and M. C. Dobson, "Classification of short vegetation using polarimetric SAR data," in *IEEE International Geoscience and Remote Sensing Symposium Proceedings*, volume 1, pp. 103–105, 1999.

[19] Y. Kouskoulas, F. Ulaby, and M. C. Dobson, "Classification of short vegetation using multifrequency SAR," in *IEEE International Geoscience and Remote Sensing Symposium Proceedings*, volume 1, pp. 103–105, 1998.

[20] J.-S. Lee, M. R. Grunes, and R. Kwok, "Classification of multi-look polarimetric SAR imagery based on complex Wishart distribution," *International Journal of Remote Sensing*, vol. 15, no. 11, pp. 2299–2311, 1994.

[21] J.-S. Lee, K. W. Hoppel, S. A. Mango, and A. R. Miller, "Intensity and phase statistics of multilook polarimetric and interferometric SAR imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 5, , September 1994.

[22] D. Lewinski, "Nonstationary probabilistic target and clutter scattering models," *IEEE Transactions on Antennas and Propagation*, vol. 31, no. 3, , May 1983.

[23] F. P. Miranda, L. Fonseca, J. R. Carr, and J. V. Taranik, "Analysis of JERS-1 (Fuyo-1) SAR data for vegetation discrimination in northwestern brazil using the semivariogram textural classifier," *International Journal of Remote Sensing*, vol. 17, no. 17, pp. 3523–3529, 1996.

[24] T. Nagai, Y. Yamaguchi, and H. Yamada, "Use of multi-polarimetric enhanced images in SIR-C/X-SAR land-cover classification," *IEICE Transactions in Communication*, vol. E80-B, no. 11, , November 1997.

[25] L. E. Pierce, F. Ulaby, K. Sarabandi, and M. C. Dobson, "Knowledge-based classification of polarimetric SAR images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 32, no. 5, , September 1994.

[26] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C, The Art of Scientific Computing*, Cambridge University Press, second edition, 1992.

[27] E. Rignot, R. Chellapa, and P. Dubois, "Unsupervised segmentation of polarimetric SAR data using the covariance matrix," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 30, no. 4, , July 1992.

[28] H. K. Ryu, "Maximum entropy estimation of density and regression functions," *Journal of Econometrics*, vol. 56, pp. 397–440, 1993.

[29] K. Sarabandi, "Derivation of phase statistics from the Mueller matrix," *Radio Science*, vol. 27, no. 5, pp. 553–560, September-October 1992.

[30] K. Sarabandi and E. Li, "Characterization of optimum polarization for multiple target discrimination using genetic algorithms," *IEEE Transactions on Antennas and Propagation*, vol. 45, no. 12, , December 1997.

[31] A. H. Schistad Solberg, T. Taxt, and A. K. Jain, "A Markov random field model for classification of multisource satellite imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 34, no. 1, , January 1996.

[32] D. W. Scott, *Multivariate Density Estimation*, Wiley, 1992.

[33] C. E. Shannon, "A mathemetical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[34] B. Silverman, *Density Estimation for Statistics and Data Analysis*, Chapman and Hall, 1986.

[35] J. Skilling and R. K. Bryan, "Maximum entropy image reconstruction: General algorithm," *Monthly Notices of the Royal Astronomical Society*, vol. 211, pp. 111–124, 1984.

[36] A. A. Swartz, H. A. Yueh, J. A. Kong, L. M. Novak, and R. T. Shin, "Optimal polarizations for achieving maximum contrast in radar images," *Journal of Geophysical Research*, vol. 93, no. B12, , December 1988.

[37] M. Tribus, "Thirty years of information theory," in *The Maximum Entropy Formalism*, MIT Press, 1979.

[38] F. T. Ulaby and C. Elachi, editors, *Radar Polarimetry for Geoscience Applications*, Artech House, Inc., 1990.

[39] F. T. Ulaby, R. K. Moore, and A. K. Fung, *Microwave Remote Sensing Active and Passive*, volume Second, pp. 476–483, Artech House, Inc., 1982.

[40] J. van Zyl, *On The Importance of Polarization In Radar Scattering Problems*, PhD thesis, California Institute of Technology, 1986.