Spectral Feature Design In High Dimensional Multispectral Data

Chih-Chien Thomas Chen
David A. Landgrebe

School of Electrical Engineering
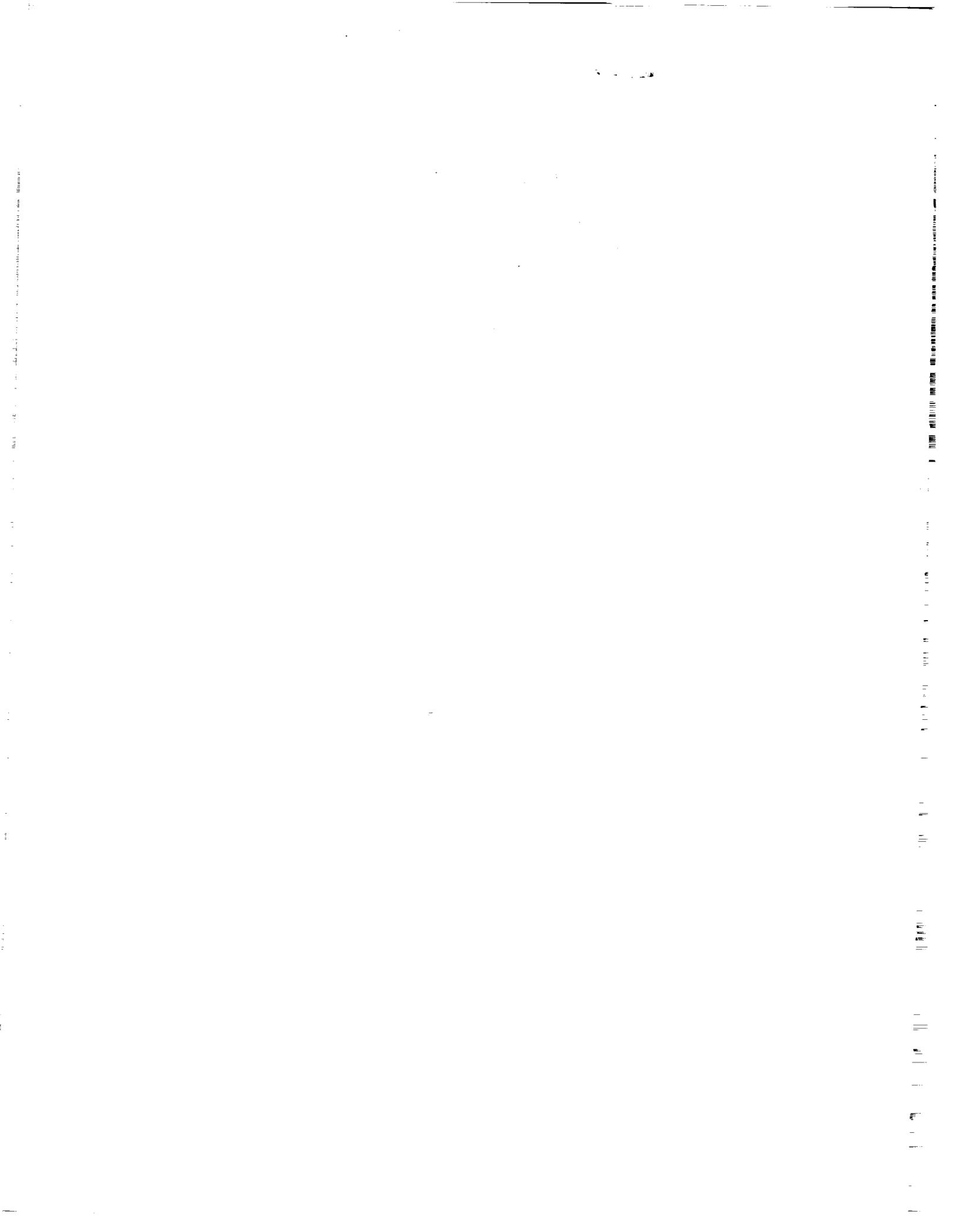Purdue University
West Lafayette, Indiana 47907

SPECTRAL FEATURE DESIGN IN HIGH DIMENSIONAL MULTISPECTRAL DATA

Chih-Chien Thomas Chen
David A. Landgrebe

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Chen, Chih-Chien Thomas   Ph.D.,   Purdue   University, August 1988.
Spectral Feature Design in High Dimensional Multispectral Data. Major
Professor :   David A. Landgrebe.   School of Electrical Engineering.

The High resolution Imaging Spectrometer (HIRIS) is designed to
acquire images simultaneously in 192 spectral bands in the 0.4-2.5 μm
wavelength region. It will make possible the collection of essentially continuous
reflectance spectra at a spectral resolution sufficient to extract significantly
enhanced amounts of information from return signals as compared to existing
systems. By effectively utilizing these signals, direct identification of the
parameters of species can be achieved and their subtle changes can also be
observed and measured.

The advantages of such high dimensional data come at a cost of
increased system and data complexity. For example, since the finer the
spectral resolution, the higher the data rate, it becomes impractical to design
the sensor to be operated continuously. Even operating HIRIS in a request only
mode, its 512 Mbps raw data rate still constitutes a serious communication
challenge. In order to solve this problem, it is essential to find new ways to
preprocess the data which reduce the data rate while at the same time
maintaining the information content of the high dimensional signal produced.

In this thesis, four spectral feature design techniques are developed from the Weighted Karhunen-Loeve Transforms. They are non-overlapping band feature selection algorithm, overlapping band feature selection algorithm, Walsh function approach, and infinite clipped optimal function approach. From a simplicity and effectiveness point of view, the infinite clipped optimal function approach is chosen since the features are easiest to find and their classification performance is the best. This technique approximates the spectral structure of the optimal features via infinite clipping and results in transform coefficients which are either +1, -1 or 0. Therefore the necessary processing can be easily implemented on-board the spacecraft by using a set of programmable adders that operate on the grouping instructions received from the ground station.

After the preprocessed data has been received at the ground station, canonical analysis is further used to find the best set of features under the criterion that maximal class separability is achieved.

In this research, both 100 dimensional vegetation data and 200 dimensional soil data are used to test the spectral feature design system. It will be shown that the infinite clipped versions of the first 16 optimal features derived from the Weighted Karhunen-Loeve Transform have excellent classification performance. Further signal processing by canonical analysis increases the compression ratio and retains the classification accuracy. The overall probability of correct classification is over 90% while providing for a reduced downlink data rate by a factor of 10.

CHAPTER I

INTRODUCTION

1.1 Research Objective

Due to the recent advance in optics and solid state technology, it is now possible to build sensors with much finer spectral resolution. This will provide the opportunity for collecting data for a much enriched information source. For example, the future High resolution Imaging Spectrometer (HIRIS) is planned to have as many as 192 spectral bands [1]. Since the signal dimensionality is tremendously increased, current techniques for analyzing multispectral data would not be adequate. In order to effectively utilize the information collected and achieve these benefits from the high dimensional measurements, it is essential to find new ways to process the data which reduce the data rate while at the same time maintaining the information content of the signals produced.

The fundamental objective of this research is to develop an objective and practical spectral feature design technique for high dimensional multispectral data.

One possible approach that might be used to accomplish the design objective is to tailor the spectral features to the particular analysis problem at hand. Features might be made up by grouping (i.e. summing) the narrow band response functions in particular spectral regions on board the spacecraft, based

upon the particular classes of ground cover parameters that are to be identified. The main advantage of this approach is the possibility of local optimality. Instead of finding optimal features with respect to all possible scenes (global optimal), a more practical and adaptive approach is introduced for each individual situation. The maximal attainable performance of local optimal features is indeed better and at least not worse, than that of global optimal ones. The problem then reduces to finding a means for deciding how to choose these band groupings effectively for each different analysis situation such that the data rate is greatly reduced while the classification performance is preserved or increased.

1.2   Previous Approaches

There have been basically four approaches to this feature design problem. They are (1) in-depth studies of physical considerations, (2) empirical methods, (3) simulation methods, and (4) analytical approaches.

Important physical considerations which have been investigated are atmospheric effects and the interaction of light with various cover types. By evaluating the transmittance of the atmosphere over the spectral interval of interest [2,3] , one can eliminate certain portions of the interval, since little or no information content is contained in those regions.

The interaction of electromagnetic radiation with plant leaves [4] , soils [5] and waters [6] has been studied in the past to find the most effective spectral features for discrimination. A typical procedure for these studies is to take

measurements with a spectroradiometer on restricted information classes over the entire spectrum. Then the average of the spectral responses is found and the subsequent conclusion is drawn from the average. The basic disadvantage of this approach is that only the mean value is considered. The potential information in the variance and covariance is neglected and lost.

The second method is empirical in that a scanner with many spectral bands is constructed, and the selection of the bands is done experimentally. The studies have been done with agriculture cover types [7] , forest covers [8], and geological applications [9]. The main advantage of the empirical method is the retaining of the information in the variations about the mean. The correlation is considered in the feature design process. However, the spectral sampling is crude and incomplete for representing the whole spectrum.

Simulation methods have been developed [10] to generate typical spectra according to a scene model. These artificial spectral response functions are then used to choose the best set of features. However, due to the complexity of the scene and the interrelations of various parameters [11], an accurate enough model of the scene is not available yet up to present.

The recent advances in optical and solid state technologies make it possible to build high dimensional multispectral sensors such as HIRIS, with a spectral resolution of 10 nm and a spatial resolution of 30m [1]. In order to effectively utilize, including acquire, archive, retrieve, transmit and analyze the data collected, analytical feature design approaches are sought because of their objective and machine-oriented natures. Early works of this approach are found in Wiswell's and Wiersma's Ph.D dissertations. Wiswell [12] studied the

feasibility of using the maximal average mutual information [13] as a criterion to evaluate the spectral features. The best set of features are chosen so as to obtain the minimal reduction in uncertainty about the scene after the observation is made. The research showed that average mutual information is a useful concept to construct the feature sets. However the relationship between average mutual information and global performance criterion such as classification accuracy was not demonstrated. Moreover, the technique was only applied to much lower dimensional signals (about 10); the feasibility for high dimensional signals in the range of one or two hundred spectral bands was not shown.

Wiersma and Landgrebe [14,15] proposed the use of minimum mean square representation error criterion for feature design. It was shown that an analytical feature design procedure can be established by applying a weighted Karhunen-Loeve Transform [16,17,18] to the observation space in which the eigenvectors of the transform are the optimal (though impractically complex) spectral features. The dimensionality in this research was 100 which was much higher than that in Wiswell's work. A manual band feature selection was suggested according to the relative importance of spectral regions as indicated by the eigenfunctions. The concept of spectral dominancy was introduced although the final stages of the feature design process were manually implemented. This appears to be tedious and impractical when the number of cover types is greatly increased. Another drawback in Wiersma's work lies basically in the subjective nature of the manual feature design process.

1.3 Current Investigation

The research results presented here will adopt some procedures to extend Wiersma's work in such a way that objective, machine implemented spectral feature design schemes become feasible. The idea of local optimality is introduced in this thesis. Instead of finding the features that are optimal with respect to all possible scenes (global optimal), it is now proposed to tailor the spectral features to the specific user problem at hand. The maximally attainable performance can then be increased. The new concept of structure similarity and its realization are discussed in this dissertation. This makes the feature design problem more general in the sense that overlapping features become practical and easily implemented.

In this research four methods are developed which in effect lead to suboptimal but now practical versions of the optimal features. These derived spectral features were obtained by combining groups of adjacent spectral samples into bands, usually one or more hundred nanometers wide, that are specially tailored to the analysis task at hand. These features could be implemented by utilizing simple programmable adders at the sensor output as shown in Figure 1.1

ON-BOARD FEATURE FORMATION SYSTEM
SCHEMATIC DIAGRAM
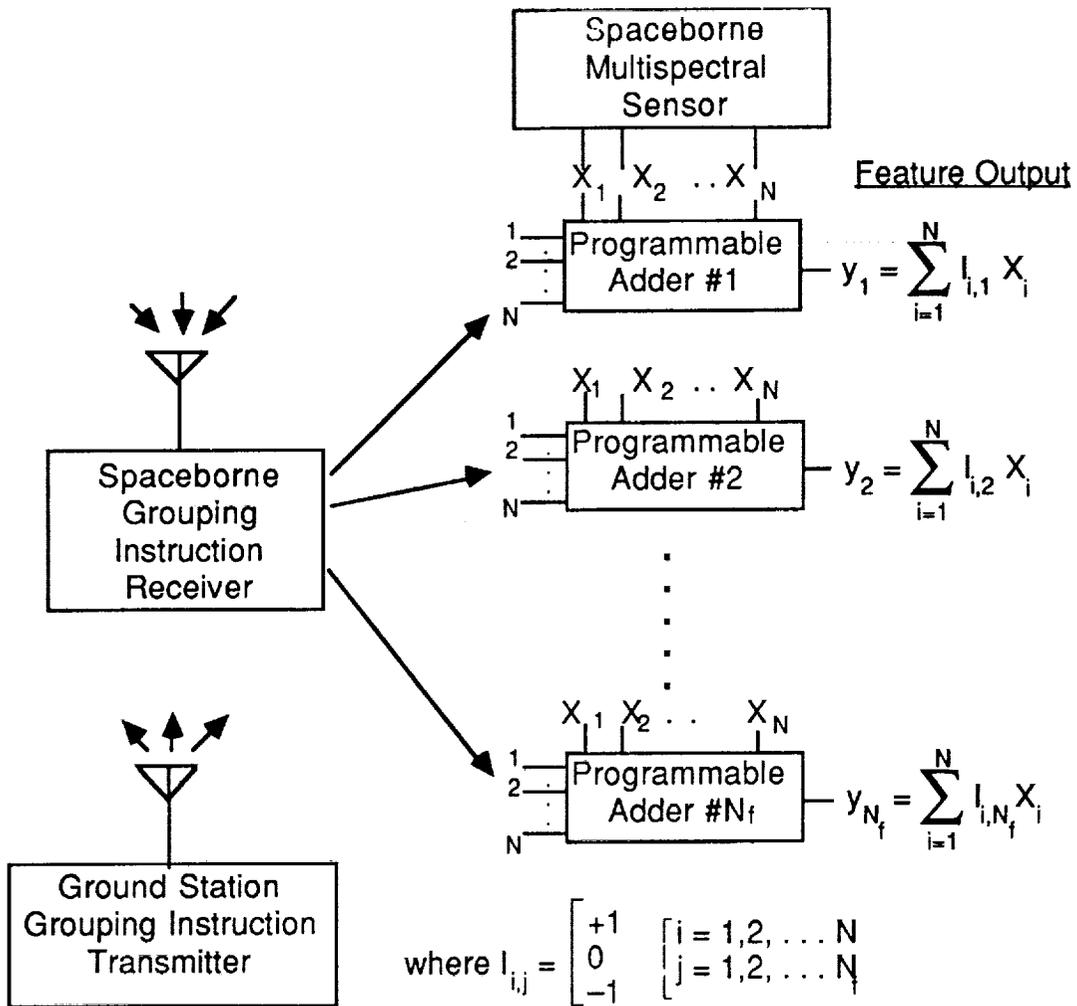
Figure 1.1 Realization of Spectral Feature Design

In Figure 1.1, N is the signal dimensionality from the sensor output, and N_f is the number of spectral features used. The programmable adders on board the spacecraft act according to the received grouping instruction from the ground station, either adding (+1), subtracting (-1) or omitting (0) bands for each spectral function. The resulting features are then transmitted down to the ground station for further processing.

The first method is based on the dominancy property of the spectral bands. A manually subjective selection process was used previously in Wiersma's work [14,15]. In this research, an objective and machine oriented process is developed. The spectral band edges are found by applying infinite clipping [21] to the average of the first few eigenvectors associated with the largest eigenvalues. This technique is referred to as a non-overlapping (N.O.L.) band feature selection algorithm due to the fact that designed features are not overlapping.

The second approach utilizes a transformation from the optimal feature space to a new space based upon Walsh Functions (W.F.) [19,20]. These functions have the attractive features of being everywhere equal to either +1 or - 1, and being ordered by the number of axis crossings. Thus the transformation can be implemented by either adding or subtracting bands, and the various functions will correspond to spectral ranges of a variety of widths.

The third scheme applies infinite clipping (I.C.) [21] to the original optimal functions derived from the weighted K-L transform. The resulting features are the infinite clipped optimal functions. In this thesis, the experiment concludes

that this scheme is the most promising technique in the sense of best classification performance under the same compression requirement.

The fourth approach extracts the zero crossing information from each optimal function and chooses those spectrum intervals that are in between two zero crossings as band features. Since the band features derived from each optimal function in this way might be linearly dependent [22], special precaution must be taken to get rid of linearly dependent bands. This method is called overlapping (O.L.) band feature selection algorithm because the bands derived by this scheme are overlapping.

1.4 Preliminary Test of the On-Board Preprocessing System

From a simplicity and effectiveness point of view, not all the four developed approaches are ideal for data preprocessing. Six preliminary test data sets are used to select the best technique. The goal is to find the most effective scheme under the simplicity requirement. Of the six sets of high spectral resolution field measurement data, three were taken over Williams County, North Dakota, each with 3 information classes: spring wheat, summer fallow and natural pasture. The other three were taken over Finney County, Kansas, again with 3 information classes each: winter wheat, summer fallow, and grain sorghum or other crops. For convenience, these data sets are referred to with a letter/number designator as shown in Table 1.1.

These data were taken by the Field Spectrometer System (FSS) [23] mounted in a helicopter. The spectral resolution was 0.02 μm for the interval from 0.4 μm to 2.4 μm.

Table 1.1 Data Set Designation for Preliminary Test

Location	Date	Designation	#of Observ.
Kansas	9/28/76	K1	832
Kansas	5/03/77	K2	1551
Kansas	6/06/77	K3	1477
N. Dakota	5/08/77	N1	1265
N. Dakota	6/29/77	N2	1239
N. Dakota	8/04/77	N3	1444

For each of the six data sets, the collection of the spectral sample functions forms the ensemble of a random process. The mean vector and the covariance matrix of this ensemble are first estimated. The estimate of the covariance matrix is used to solve the generalized Karhunen-Loeve equation which results in the eigenvalues and the eigenvectors of the transform. Figure 1.2 shows the magnitude of the first 12 eigenvectors associated with the largest eigenvalues for the data set K2 [15]. They will be used to explain the feature design schemes in chapter III. The spectral interval is 0.02 μm as stated previously. Therefore the dimensionality used in these preliminary tests is 100.

From this preliminary test, it is concluded that the infinite clipped optimal transform is the simplest and most effective method for on-board data preprocessing.

Eigenfunction 1

Eigenfunction 2

Eigenfunction 3

Eigenfunction 4

Eigenfunction 5

Eigenfunction 6

Wavelength - μm

Wavelength - μm
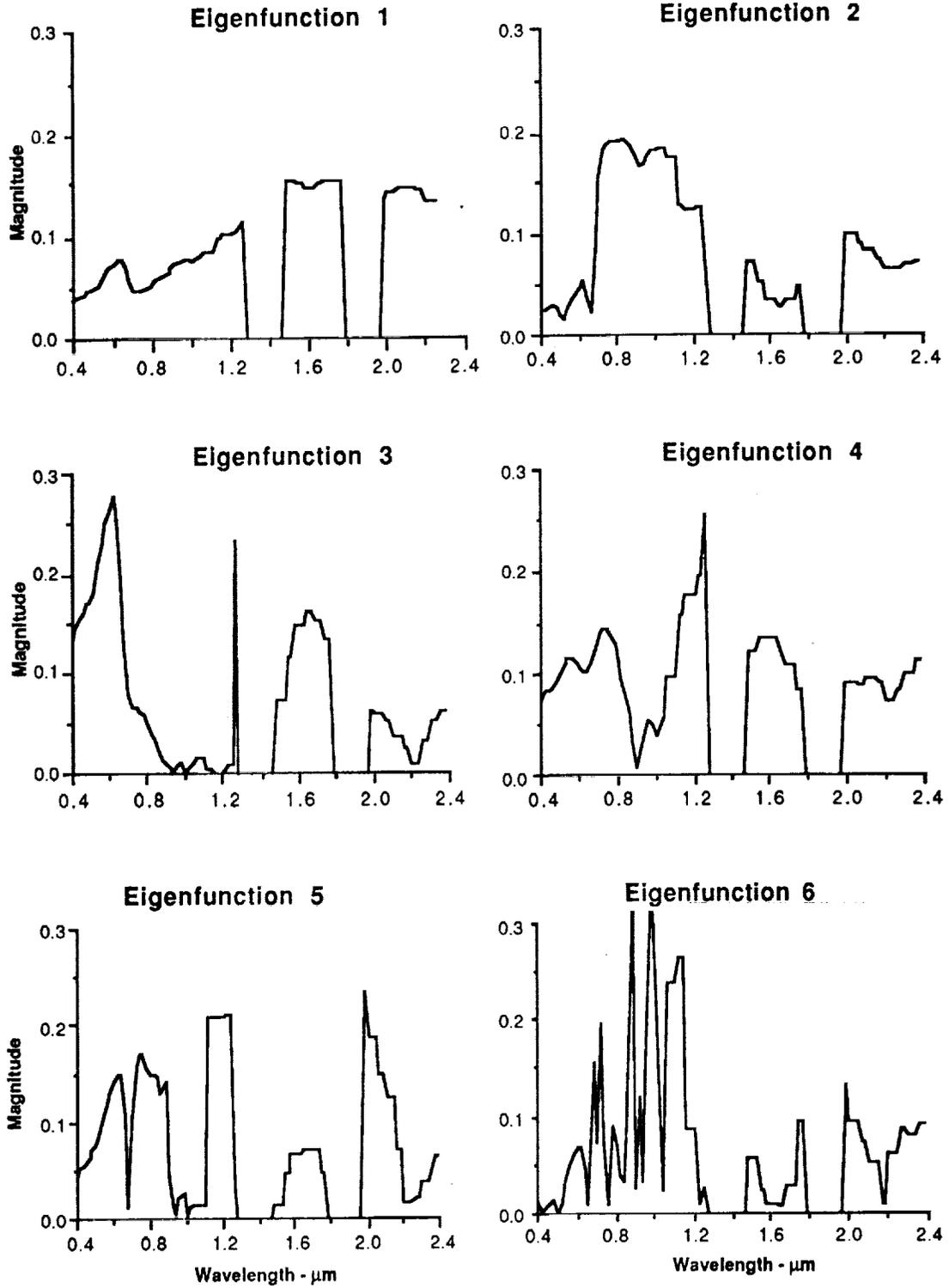
Figure 1.2   First 12 Eigenvectors of Data Set K2
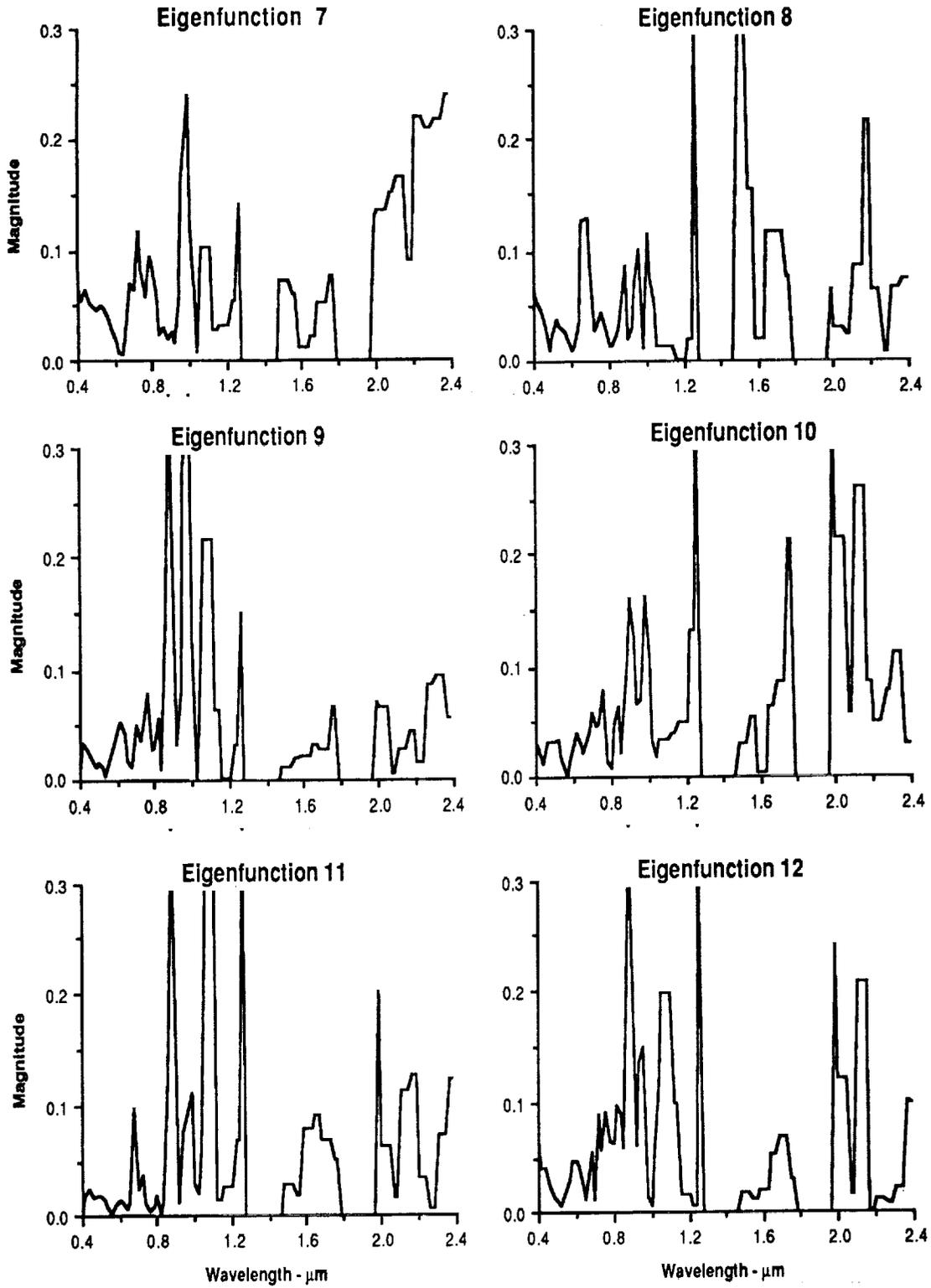
Figure 1.2, continued

1.5 Outline of the Thesis

In chapter 2, a theoretical review of the weighted K-L transform is given. Two important properties, minimum mean square truncation error and uncorrelated transformed coefficients are proved for this generalized transform.

Chapter 3 discusses in detail the four schemes developed to design the spectral features in high dimensional multispectral data. Two of them, non-overlapping band feature selection algorithm and overlapping band feature selection algorithm, are developed from the dominancy concept in eigenfunctions; and the other two, Walsh function approach and infinite clipped optimal function approach are derived from the idea of structure similarity between two sets of functions. Furthermore, a comparison among these data preprocessing schemes is included in this chapter. From the simplicity and effectiveness point of view, it is found that the infinite clipped optimal function approach is the best technique. After the preprocessed data would be received at the ground station, canonical analysis would be applied to the infinite clipped optimal transformed data to obtain maximal class separability.

Chapter 4 shows the final results of this research. Both the vegetation data and the soil data are included in this chapter. The Hughes phenomenon is also discussed.

Chapter 5 summaries the final conclusions and gives recommendations for the future work.

An IBM 3083 Macro file used to run the spectral feature design system and the source code of the system are given in the appendices.

CHAPTER II

KARHUNEN-LOEVE TRANSFORM

The Karhunen-Loeve (KL) expansion [44] was developed to represent random processes. It maps the continuous parameter random process into a sequence of random variables [24]. The expansion generates a set of deterministic orthonormal basis functions. This set has a unique error-minimizing property and uncorrelated transformed coefficients. These properties make it the optimal coordinate system for many feature design problems.

This transform can be generalized [25,26] to include a weighting function to account for certain types of a priori knowledge of the parameter set, and its proper use may have an important impact on the extraction of useful information [15]. Thus using the weighted form of K-L transform may result in more practical and realizable feature design.

In the following we will show that minimum mean square truncation error (MMSE) and uncorrelated coefficients properties, which are directly related to this research, also hold for the generalized K-L transform. The MMSE property ensures that the eigenfunctions associated with the largest eigenvalues derived from the weighted K-L transform are the optimal basis functions in the sense of signal representation. Uncorrelated coefficients property guarantees that the transformed coordinates are independent under Gaussian assumption.

2.1 Minimum Mean Square Truncation Error

Let $X(\lambda)$ be a sample function of a random process . Assume that the random process is continuous in probability and almost every sample function of the random process has finite norm in $L_2(\Lambda)$ space [27]. Then $X(\lambda)$ can be represented by an expansion of the form [24]

$$X(\lambda) = \sum_{i=1}^{\infty} y_i \; \Phi_i (\lambda) \tag{2.1}$$

where the functions $\{\Phi_i (\lambda)\}$ are deterministic and the expansion coefficients $\{ y_i \}$ are random variables.

Define a weighting function $W(\lambda)$ with real finite positive values. Without loss of generality, the set $\{\Phi_i (\lambda)\}$ will be taken to be orthonormal with respect to $W(\lambda)$. From the generalized inner product [27] which defines the metric in $L_2(\Lambda)$ space, we have

$$(\Phi_i , \; \Phi_j)_W = \int_{\Lambda} \Phi_i (\lambda) \, W (\lambda) \, \Phi_j (\lambda) \, d\lambda = \begin{cases} 0 \text{ if } i \neq j \\ 1 \text{ if } i = j \end{cases} \tag{2.2}$$

and

$$y_i = (\Phi_i , X)_W = \int_{\Lambda} \Phi_i (\lambda) \, W (\lambda) \, X (\lambda) \, d\lambda \tag{2.3}$$

If the set $\{\Phi_i(\lambda)\}$ is not orthonormal to begin with, it can be orthonormalized by the Gram-Schmidt procedure [57]. That such sets exists in $L_2(\Lambda)$ space has been demonstrated by the construction of sets such as

complex sinusoids, Legendre polymonials, Chebyshev polymonials, Laguerre functions, Walsh functions and others.

Therefore $Y = \{ y_1, y_2, \dots \}$ is simply an orthonormal transformation of the random function $X(\lambda)$, and is itself a random vector. Each component of Y is a feature which contributes to representing the observed sample function $X(\lambda)$.

Furthermore, we are going to choose a set $\{\Phi_i(\lambda)\}$ which is complete in $L_2(\Lambda)$ space. That is, if we define the sequence

$$c_n(\lambda) = \sum_{i=1}^{n} y_i \, \Phi_i(\lambda) \qquad (2.4)$$

then,

$$\lim_{n \to \infty} \left\{ \int_{\Lambda} [\, X(\lambda) - \sum_{i=1}^{n} y_i \, \Phi_i(\lambda) \,]^2 \, W(\lambda) \, d\lambda \right\} = 0 \qquad (2.5)$$

That the sequence $c_n(\lambda)$ converges to $X(\lambda)$ in the mean square sense, is denoted by

$$X(\lambda) = \lim_{n \to \infty} c_n(\lambda) \qquad (2.6)$$

This convergence guarantees that the series can be made arbitrarily close to $X(\lambda)$ by increasing n in the expansion.

The problem of designing the optimal sensor then becomes that of selecting the set of complete orthonormal (CON) basis functions { $\Phi_i(\lambda)$ } such that the series representation will be optimal with respect to the minimum mean square error criterion. In the stochastic environment, this representation error is taken over the ensemble of the random process. Hence, we need :

$$E\left\{ \int_\Lambda [X(\lambda) - \sum_{i=1}^{\infty} y_i \, \Phi_i(\lambda)]^2 W(\lambda) \, d\lambda \right\} = 0 \qquad (2.7)$$

Another desirable property is that the convergence be rapid in the first few terms, that is, each additional term used in the series expansion decreases the representation error by a maximum amount . This property is called energy packing.

In the real applications, however, it is impractical to transmit an infinite or even a very large number of channels to the ground. Therefore only a finite number of terms in the expansion would be used. Let n be a finite number such that the representation error by using the first n terms in the expansion is less than T, the maximal acceptable error. Then we require the selected orthonormal basis functions { $\Phi_i(\lambda)$ } to be complete in a finite n dimensional subspace of $L_2(\Lambda)$. That is, for any T > 0, there is an n_0 such that

$$E\left\{ \int_\Lambda [X(\lambda) - \sum_{i=1}^{n} y_i \, \Phi_i(\lambda)]^2 W(\lambda) \, d\lambda \right\} < T \quad ; n > n_0 \qquad (2.8)$$

for any $X(\lambda)$ defined in the $L_2(\Lambda)$ space.

This completeness property in finite dimensional space can guarantee that if we use the n dimensional subspace of $L_2(\Lambda)$, spanned by the first n elements of a complete orthonormal set $\{ \Phi_i(\lambda) \}$, for the representation of an arbitrary signal, then the norm of the error can be made arbitrarily small by choosing n sufficiently large.

The objective then is to find the a finite set of orthonormal basis functions that have the above minimum representation error and energy packing properties. In the following, we are going to show that the eigenfunctions derived from the Weighted Karhunen-Loeve transform are just the desired optimal basis functions.

In the above finite n dimensional subspace of $L_2(\Lambda)$, suppose only m terms in the expansion will be used to estimate the observed $X(\lambda)$, then the estimate $\hat{X}(\lambda)$ can be expressed in the following form

$$\hat{X}(\lambda) = \sum_{i=1}^{m} y_i \Phi_i(\lambda) + \sum_{i=m+1}^{n} b_i \Phi_i(\lambda) \tag{2.9}$$

The constants $\{ b_i \}$ are preselected. The objective is to find the basis functions and the constants $\{ b_i \}$ in such a way that the minimum mean square error can be obtained.

Since we do not use all of the basis functions, the representation error due to truncation is then equal to

$$\Delta X(\lambda) = X(\lambda) - \hat{X}(\lambda) = \sum_{i=m+1}^{n} (y_i - b_i) \Phi_i(\lambda) \tag{2.10}$$

We define the weighted mean square error to be

$$\mathbf{WMSE} = E((\Delta X, \Delta X)_W) = E\left(\sum_{i=m+1}^{n}(y_i - b_i)\sum_{j=m+1}^{n}(y_j - b_j)\int_\Lambda \Phi_i(\lambda)W(\lambda)\Phi_j(\lambda)d\lambda\right) \quad (2.11)$$

Since the basis functions are orthonormal, Eq (2.11) reduces to

$$\mathbf{WMSE} = \sum_{i=m+1}^{n} E(y_i - b_i)^2 \quad (2.12)$$

The mean square error is minimized when

$$\frac{\partial E(y_i - b_i)^2}{\partial b_i} = -2E(y_i - b_i) = 0 \quad (2.13)$$

That is, the preselected constant b_i must be equal to the expected value of the transform component $E(y_i)$.

We are left to show that when $\Phi_i(\lambda)$ is a weighted K-L basis, then the weighted mean square error is minimized. We need to minimize

$$\mathbf{WMSE} = \sum_{i=m+1}^{n} E(y_i - E(y_i))^2 = \sum_{i=m+1}^{n} \int_\Lambda\int_\Lambda \Phi_i(\lambda)W(\lambda)K_x(\lambda,u)W(u)\Phi_i(u)du\,d\lambda \quad (2.14)$$

where $K_x(\lambda,u)$ is the covariance function of the random process.

Using the orthonormality constraint, we can write the mean square error as the quadratic functional [19] of $\Phi_i(\lambda)$

$$\text{WMSE} = \sum_{i=m+1}^{n} \int_{\Lambda} \int_{\Lambda} \Phi_i(\lambda) \, W(\lambda) \, K_x(\lambda, u) W(u) \, \Phi_i(u) \, du \, d\lambda$$

$$- \sum_{i=m+1}^{n} \lambda_i \left\{ \int_{\Lambda} \Phi_i(\lambda) \, W(\lambda) \, \Phi_i(\lambda) \, d\lambda - 1 \right\} \qquad (2.15)$$

Minimizing with respect to Φ_i yields [19]

$$\nabla_{\Phi_i}(\text{WMSE}) = 2 \int_{\Lambda} W(\lambda) K_x(\lambda, u) W(u) \Phi_i(u) du - 2\lambda_i W(\lambda) \Phi_i(\lambda) = 0 \quad (2.16)$$

The set $\{\lambda_i\}$ thus turns out to be the eigenvalues of the covariance function of the observed $X(\lambda)$, and the basis functions satisfy the weighted K-L equation

$$\int_{\Lambda} K_x(\lambda, u) W(u) \Phi_i(u) du = \lambda_i \, \Phi_i(\lambda) \qquad i = 1, 2, \ldots, n \qquad (2.17)$$

From equations 2.14 and 2.17, we have

$$\text{WMSE} = \sum_{i=m+1}^{n} \int_{\Lambda} \Phi_i(\lambda) W(\lambda) \left[\lambda_i \, \Phi_i(\lambda) \right] d\lambda \qquad (2.18)$$

or

$$\text{WMSE} = \sum_{i=m+1}^{n} \lambda_i \qquad (2.19)$$

If we rank the optimal functions according to the magnitudes of their associated eigenvalues from the largest to the smallest, then using the first few optimal functions in the series representation will results in the desired weighted minimum mean square error. Furthermore, the energy packing property will also be satisfied since the mean square error reduction for using each additional term in the expansion will be maximized.

2.2   Uncorrelated Transformed Coefficients

The generalized K-L transform results in uncorrelated coefficients. This property can be derived as follows.   Since

$$Y = \{ \; y_1, \; y_2, \; ....., \; y_n \; \}$$

(2.20)

where

$$y_i = \int_\Lambda \Phi_i(\lambda) \; W(\lambda) \; X(\lambda) \; d\lambda$$

(2.21)

and the covariance between y_i and y_j is defined as

$$\sigma_{i,j} = E(y_i - E(y_i))(y_j - E(y_j))$$

(2.22)

Using Eq.(2.21), Eq.(2.22) becomes

$$\sigma_{i,j} = \iint\limits_{\Lambda\Lambda} \Phi_i(\lambda)W(\lambda)K_x(\lambda,u)W(u)\Phi_j(u)\ du\ d\lambda \qquad (2.23)$$

From the Weighted Karhunen-Loeve Equation derived in Eq.(2.17), we get

$$\sigma_{i,j} = \int\limits_{\Lambda} \Phi_i(\lambda)\ W(\lambda)\ [\ \lambda_j\ \Phi_i(\lambda)\ ]\ d\lambda = \left\{ \begin{array}{ll} \lambda_i & \text{if } i = j \\ 0 & \text{if } i \neq j \end{array} \right. \qquad (2.24)$$

Therefore the transformed coefficients are uncorrelated. If the underlying distribution of the random process is Gaussian, the coefficients are then independent.

CHAPTER III

SPECTRAL FEATURE DESIGN

From the discussion in chapter 2, we know the weighted K-L transform preserves the minimum weighted mean square error (MWMSE) and ordered uncorrelated coefficients properties. In fact, the K-L transform is a special case of its generalized form with unity weight matrix. The fundamentals in remote sensing indicate [14,15] that the eigenfunctions derived in the K-L transform with unity weight matrix can not be used satisfactorily for feature design. The reason for this is basically the fact that the reflectance around the two water absorption bands has high variance and thus tends to dominate the formation of the basis functions. Therefore the spectral response in these two regions is not information-bearing. Indeed, the spectral radiance emanates mostly from the atmosphere and must be considered as noise. Understanding this important a priori knowledge about the scene, we can incorporate a weighting function into the calculation process to eliminate the effect of noise. The generalized K-L transform is then the solution. The resulting optimal functions can be used to transform the original observation space into a new feature space.

In this chapter, four spectral feature design techniques will be presented first. Using simplicity and effectiveness as criteria, the most promising technique is then selected from these four schemes for our final feature design system. The four techniques developed in the course of this research are

1. Non-overlapping band feature selection algorithm,

2. Walsh function approach,

3. Infinite clipped optimal function approach, and

4. Overlapping band feature selection algorithm.

The non-overlapping and overlapping band feature selection algorithms are derived from the shape of the optimal features. The Walsh function approach and the infinite clipped optimal function approach are developed from the structure of the optimal features.

After performing the generalized K-L transformation to the data [15], where a weight function is incorporated into the transform to avoid portions of the spectrum where the atmosphere is known to be opaque, the eigenfunctions can be found. These eigenfunctions serve as optimal features that linearly transform the original measurement space to the new space in a minimum mean square error sense [18]. However, because of the inherently complex nature of the optimal functions, an easy and fast implementation directly using them to process the tremendous amount of information collected must be found. Therefore, more realistic features are sought in order to achieve this requirement. More realistic features can be found by carefully studying the shapes of the first few eigenfunctions. The importance of a wavelength region for the purpose of accurately representing the ensemble of functions is indicated by the magnitude of the eigenfunctions in that region. It is hypothesized that the importance of a region in an ensemble-representational sense is positively correlated with (though not identical to) its importance with respect to classification accuracy. Referring to Figure 1.2, it is observed that each eigenfunction thus points to the more important regions.

For instance, the magnitude of the first eigenfunction indicates that there are 3 important regions over the entire spectrum: 0.4-1.28 μm, 1.48-1.78 μm and 1.98-2.4 μm, the magnitude of the second eigenfunction indicates that important regions are approximately 0.4-0.66 μm, 0.66-1.28 μm, 1.48-1.78 μm and 1.98-2.4 μm, etc. From the fact that the magnitude of the first eigenfunction is very similar to the soil response function, and the magnitude of the second eigenfunction is similar to the vegetation curve, it is observed that the dominant portion of the ensemble, i.e. summer fallow , winter wheat and unknown crops for this data set K2, can be shown in the first few eigenfunctions derived from the weighted K-L transform. Therefore, it is desired to choose the regions with larger magnitude in the eigenfunctions, especially from those with largest eigenvalues, as sensor bands since these regions contribute most to reduction of representation error as well as increasing of classification performance.

However, such a subjective process is difficult to carry out objectively due to the spectral detail in the eigenfunctions and the number of eigenfunctions to be examined. A machine implemented band selection algorithm based on this dominancy concept in the eigenfunctions is thus sought.

3.1 Non-Overlapping Band Feature Selection Algorithm

Infinite clipping is a procedure used to transform the signal into its signed form [21]. There is evidence in various circumstances that the axis crossings of a signal carry a substantial portion of the information that the signal carries. For example, in the field of speech recognition [28-33], the infinite clipping procedure can been used to extract zero crossing information and perform

speech recovery very successfully. For example, Ewing and Taylor [29] showed that zero-crossing information from a speech signal is a feasible way for computer speech recognition; and Niederjohn, et al [30] showed that the set of zero-crossings of a speech waveform represents a nearly minimal set of informational attributes in the sense that any reordering or averaging of the zero-crossing intervals has a detrimental effect upon speech intelligibility.

Some other examples of using zero-crossing information of a signal can also be found in the fields of radar target detection [51-52], biomedical engineering [53], communications [54-55] and image processing [56]. Rainal [52] described a zero-crossing principle for detecting weak narrow-band signals immersed in Gaussian noise. An application of the zero-crossing principle to the detection problem of a stationary radar target in clutter was discussed. Masuda, et al [53] demonstrated in a biomedical context that the muscle fiber conduction velocity, which is known to be an index of the degree of muscle fatigue or muscle disease, can be accurately measured by using zero-crossing information from a surface electromyogram signal. In conventional communications, Voelcker [54] showed that an angle-modulated signal can be demodulated given only its zero-crossings; Wiley, et al [55] proposed an iterative demodulation procedure for very wide-band FM by use of a zero-crossing discriminator. Haralick [56] showed that the zero-crossing of second directional derivatives within the pixel's area can be used to detect the step edges in the image.

Thus, one possible approach to finding the desired procedure would be to apply infinite clipping to extract the zero crossing information. The input to this algorithm will be the average of the first few eigenfunctions. The output of

this algorithm is to be the band edges showing how the bands should be chosen. We will refer to this procedure as the non-overlapping (N.O.L.) band feature selection algorithm. Figure 3.1 shows the average of the first 12 eigenfunctions. After thresholding, the data of Figure 3.1 become as in Figure 3.2 where +1 represents the positive portions of Figure 3.1, -1 represents the negative portions, and 0 represents the water absorption bands centered at 1.4 and 1.9 μm respectively. It should be noted that there is no response over the above water absorption bands due to the use of the weight function in the K-L transform, which has been set 1.0 over the entire spectrum and a very small positive value in the water bands.

The band edges are found as follows: whenever a transition in sign or magnitude occurs in Figure 3.2, the wavelength of the associated band is recorded. Table 3.1 shows the results after transition operation. The band edges in Table 3.1 can be used to set up the suboptimal basis functions for data compression [refer to the 2nd column in Table 3.6].

Table 3.1. Band Edges Obtained by Infinite Clipping of the Average of the First 12 Eigenvectors for Data Set K2

Band	wavelength (μm)
1	0.40 - 0.68
2	0.68 - 0.90
3	0.90 - 0.92
4	0.92 - 0.94
5	0.94 - 1.00
6	1.00 - 1.06
7	1.06 - 1.12
8	1.12 - 1.26
9	1.26 - 1.28
10	1.48 - 1.78
11	1.98 - 2.40
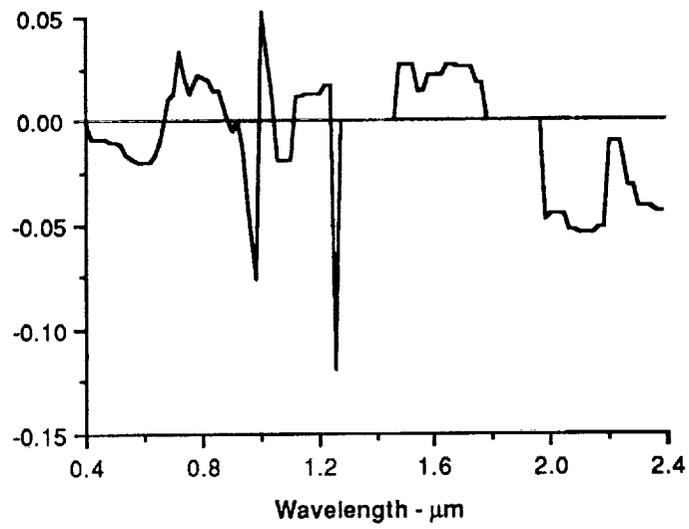
Figure 3.1   Average of the First 12 Eigenvectors of Data Set K2
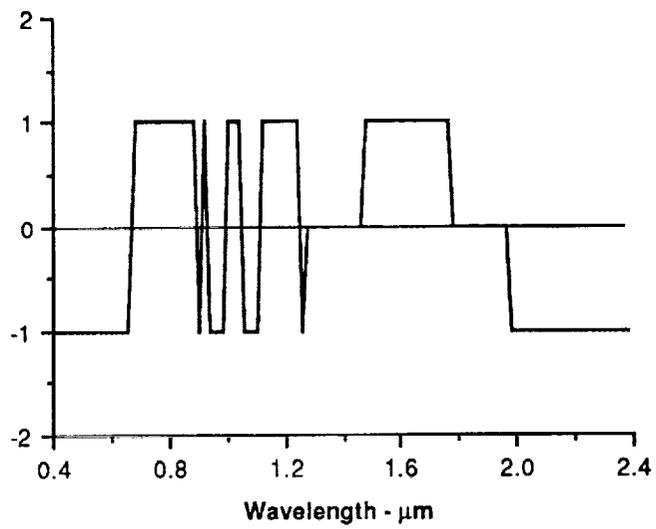
Figure 3.2   Thresholded Version of Figure 3.1

3.2 Walsh Function Approach

By carefully viewing the structure of the eigenfunctions in Figures 1.2, one may also observed that the eigenfunctions corresponding to the larger eigenvalues tend to have coarser structure than those with smaller eigenvalues. A similar effect exists in the Walsh functions indexed by the number of zero-crossings. The higher the index of the Walsh function, the finer the structure of the function [19,20]. The first 10 Walsh functions indexed by the number of axis crossings are shown in Figure 3.3, where curve 0 is the first Walsh function with no axis crossing, curve 1 is the second Walsh function with one axis crossing, etc.

The inner product of the two functions may be thought of as a mathematical measure of similarity of the two functions. The absolute values of the inner products of the first 16 eigenfunctions with the first 64 Walsh functions are calculated. Table 3.2 shows part of the results. Absolute values of the inner product are used since the polarity is not significant here. Table 3.3 shows the similarity relation between these two sets of functions. For example, the number "1" in the (1,1) matrix position indicates that the first eigenfunction is more similar to the first Walsh function than to any other 63 Walsh functions since the value 0.84 in Table 3.2 is the largest in the " first " column. The numbers "2", "3" and "4" in the (1,2), (1,3) and (1,4) matrix positions indicate that the 2nd, 3rd and 4th eigenfunctions mostly look like the 2nd, 3rd and 4th Walsh functions respectively in the sense of signal structure similarity. Therefore, the structure of the first 4 eigenfunctions can be approximated by that of the first 4 Walsh functions. By observing the first two rows of Table 3.3, it can be concluded that the first 16 eigenfunctions and the first 16 Walsh functions have approximately

the same structure. The structure in the eigenfunctions is related to the axis crossings in the signals. The coarser the structure, the less the number of axis crossings; and vice versa. These axis crossings are hypothesized to contain important information that can be used for classification. Therefore, it is feasible to use the first few Walsh functions as spectral features in high dimensional multispectral data.
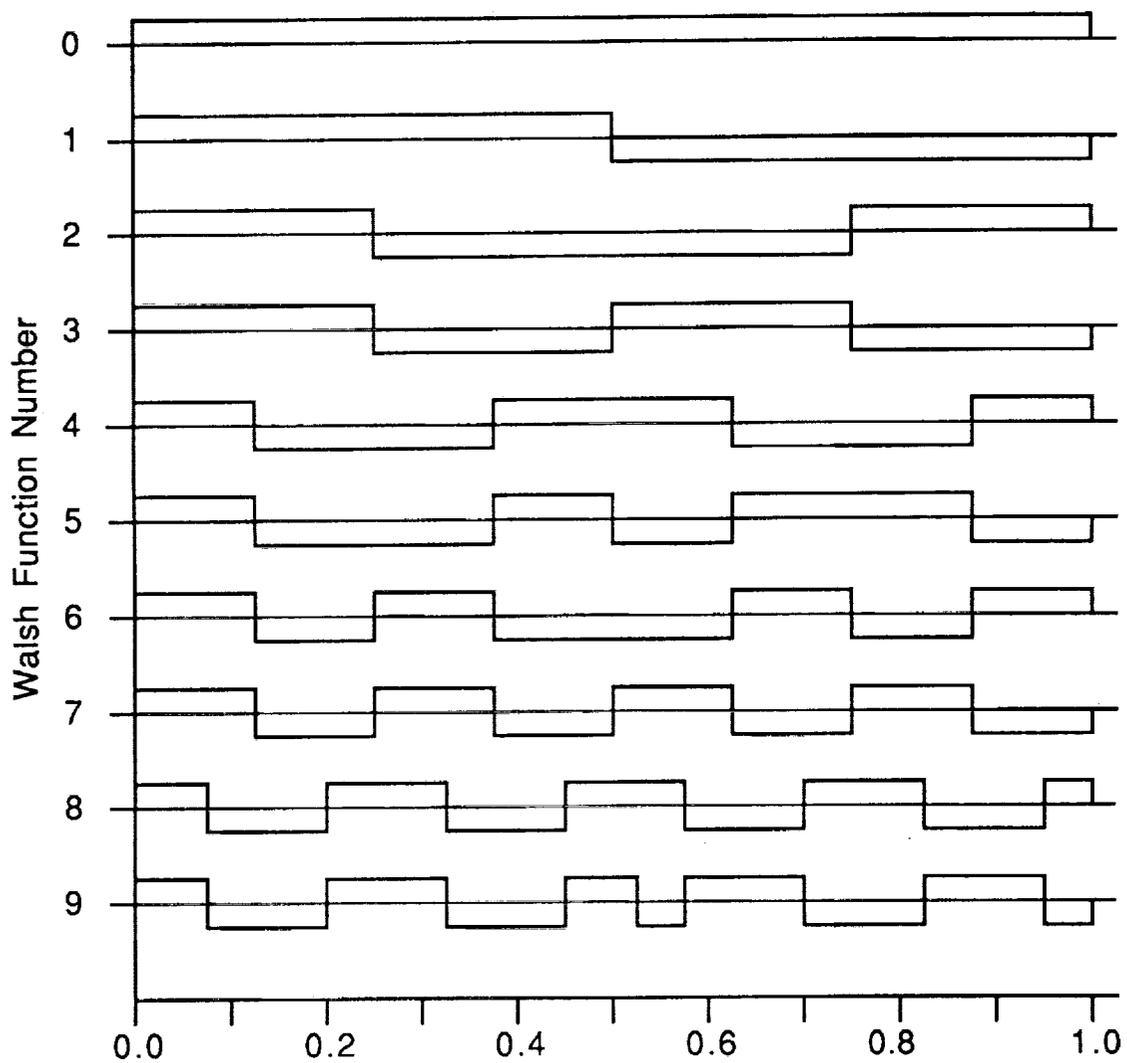
**Figure 3.3 First 10 Walsh Functions Indexed
By Number of Axis Crossings**

Table 3.2   Absolute Values of Inner Products Between Optimal Functions and Walsh Functions

Optimal#	1	2	3	4	5	6	7	8
Walsh#								
1	**0.84**	0.21	0.21	0.09	0.01	0.01	0.00	0.00
2	0.21	**0.68**	0.42	0.12	0.24	0.01	0.13	0.03
3	0.04	0.23	**0.66**	0.05	**0.43**	0.02	0.17	0.17
4	0.09	0.03	0.09	**0.78**	0.12	0.01	0.03	0.09
5	0.04	0.39	0.13	0.05	0.40	0.03	0.13	0.17
6	0.11	0.32	0.09	0.01	0.28	0.14	0.33	0.25
7	0.06	0.11	0.13	0.09	0.20	**0.35**	0.23	0.10
8	0.03	0.10	0.15	0.06	0.03	0.03	**0.52**	**0.36**
9	0.25	0.07	0.03	0.05	0.29	0.14	0.16	0.28
10	0.12	0.05	0.26	0.24	0.27	0.02	0.20	0.14
11	0.13	0.15	0.21	0.06	0.15	0.08	0.18	0.15
12	0.03	0.15	0.05	0.32	0.09	0.07	0.21	0.02
13	0.02	0.18	0.00	0.04	0.08	0.00	0.09	0.03
14	0.15	0.10	0.04	0.15	0.00	0.07	0.09	0.10
15	0.08	0.03	0.09	0.16	0.09	0.15	0.01	0.14
16	0.03	0.04	0.03	0.04	0.20	0.18	0.05	0.10

Optimal#	9	10	11	12	13	14	15	16
Walsh#								
1	0.01	0.00	0.01	0.01	0.03	0.04	0.01	0.05
2	0.00	0.03	0.08	0.08	0.07	0.04	0.02	0.06
3	0.00	0.04	0.06	0.00	0.07	0.02	0.06	0.09
4	0.12	0.18	0.18	0.14	0.06	0.04	0.02	0.00
5	0.13	0.21	0.02	0.09	0.04	0.14	0.14	0.19
6	0.09	0.24	0.03	0.07	**0.29**	0.02	0.16	0.16
7	0.07	0.09	0.05	0.10	0.02	**0.39**	0.23	0.03
8	0.03	0.05	0.10	0.03	0.06	0.15	0.15	0.13
9	0.22	0.06	0.08	**0.29**	0.21	0.19	0.13	0.07
10	0.07	0.10	0.32	0.00	0.06	0.27	0.12	0.17
11	0.14	0.16	0.08	0.10	0.14	0.07	0.01	**0.40**
12	0.21	0.00	0.05	0.23	0.08	0.14	0.11	0.16
13	0.01	0.11	**0.33**	0.19	0.13	0.08	0.00	0.09
14	0.11	0.00	0.24	0.12	0.08	0.10	0.07	0.08
15	**0.27**	0.06	0.06	0.05	0.18	0.07	0.05	0.04
16	0.12	0.24	0.19	0.01	0.01	0.07	0.02	0.05

Table 3.3 Similarity Relation Between Optimal Functions and Walsh Functions

Optimal#	1	2	3	4	5	6	7	8
Rank	Walsh#							
1	1	2	3	4	3	7	8	8
2	57	5	2	12	5	36	6	9
3	9	6	59	60	9	16	7	6
4	2	3	10	10	6	40	28	22
5	14	1	11	15	10	35	12	18
6	11	58	1	14	2	19	10	24
7	10	13	58	2	7	23	25	64
8	33	11	8	52	16	15	11	3
9	6	12	27	29	21	63	3	5
10	58	59	7	7	59	9	24	50
11	47	7	5	35	58	32	9	11
12	4	42	50	1	11	6	64	19
13	25	8	35	50	26	20	30	25
14	15	14	4	17	28	54	2	36
15	42	25	15	43	45	47	19	10
16	18	18	26	57	49	57	5	17

Optimal#	9	10	11	12	13	14	15	16
Rank	Walsh#							
1	15	17	13	9	6	7	20	11
2	22	16	10	22	23	10	24	19
3	9	6	14	12	19	21	18	20
4	12	21	26	62	9	9	7	5
5	18	5	58	54	28	33	19	36
6	50	19	16	52	15	8	52	27
7	17	53	49	13	34	5	6	50
8	49	20	38	47	29	12	8	51
9	54	4	4	43	38	49	33	10
10	26	51	55	20	11	42	5	6
11	11	43	23	34	13	31	9	12
12	36	42	44	18	25	29	30	22
13	5	11	17	50	55	28	10	34
14	29	18	31	61	52	18	50	52
15	16	61	37	30	62	14	12	8
16	35	49	63	33	44	46	44	18

3.3   Infinite Clipped Optimal Function Approach

If one studies the Walsh functions more carefully , it is found that although the Walsh functions approximate the optimal functions in the sense of structure similarity, they do distort some of the spectral spacing information in the optimal functions. The axis crossing separation in the optimal functions is a relatively irregular pattern, while it is quite regular in the Walsh functions.

One way that can be applied to avoid this information loss is to use the infinite clipped optimal functions as spectral features. The infinite clipped optimal function approach preserves the zero-crossing information in the optimal functions which is hypothesized to contain important spectral information that can be used for classification.

Furthermore, the Walsh function approach is less flexible than the infinite clipped optimal function approach since the spectral features using the Walsh functions tend to be fixed for all analysis situations; while, on the other hand, the infinite clipped optimal function approach does give some degree of adaptability. Figure 3.4 shows the infinite clipping versions of the first 6 eigenfunctions for data set K2.

The infinite clipped optimal functions, derived from the signs of the optimal functions, are then used as spectral features (i.e., basis functions) to linearly transform the high dimensional multispectral data to the ground station for further processing.
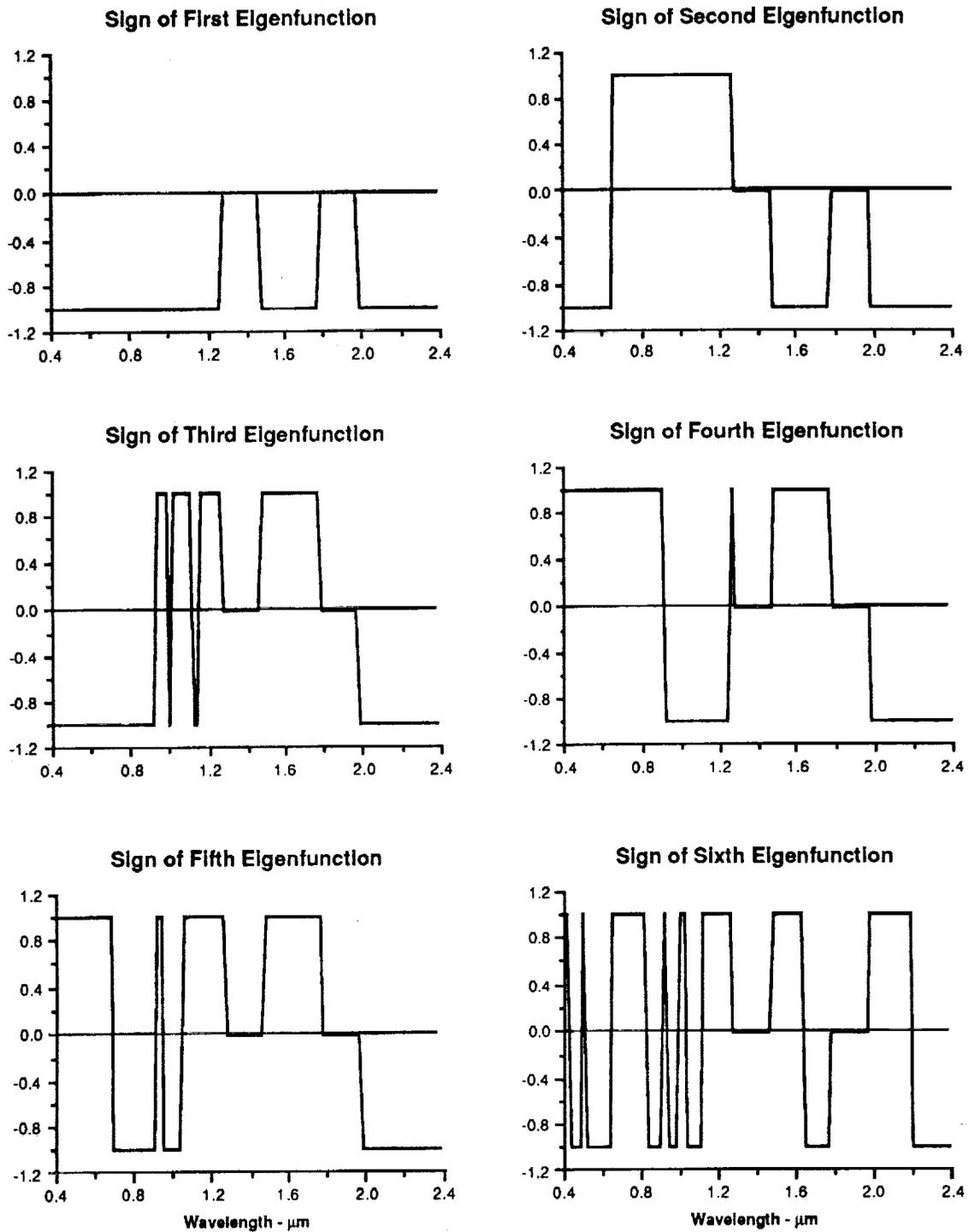
Figure 3.4   Infinite Clipping Versions of the First 6
Eigenfunctions for K2

3.4 Overlapping Band Feature Selection Algorithm

The overlapping band feature selection algorithm originates from the inherent overlapping property of the optimal functions. This property suggests that overlapping bands might be even more powerful for spectral feature design. The idea of this algorithm is to find the locations of the important spectral bands without imposing the additional restriction that the bands be non-overlapping. The basic procedures used are very similar to those in the non-overlapping band feature selection algorithm. In the non-overlapping band feature selection algorithm, the infinite clipping procedure is applied to the average of the first few eigenfunctions in order to extract the information of the important spectral bands; while in this overlapping case, the infinite clipping procedure is applied to each individual eigenfunction.

The first step is to find the band edges of each individual eigenfunction. Table 3.4 shows part of the results for data set K2. In Table 3.4, comparing to Figure 1.2, it is found that there are 3 important bands for the first eigenfunction, 4 for the 2nd one, 8 for the 3rd one, etc.

It should be noted that the band features derived in this way are not all linearly independent. For example, the first and second band feature from the second eigenfunction, that is, 0.40-0.66 μm and 0.66-1.28 μm, are linearly dependent on the first band feature from the first eigenfunction (0.40-1.28 μm). Another example is the identical band features (1.48-1.78 and 1.98-2.40 μm) derived from the first 5 eigenfunctions. Indeed, these repeated bands and the bands which are linearly dependent on the previously selected bands can not

be used as spectral features since linearly dependent features will result in singular class covariance matrix.

Table 3.4 Linearly Dependent Bands Found by Overlapping Band Feature Selection Algorithm for Data Set K2

EigenVector#	1	2	3
BAND			
1	0.40 - 1.28	0.40 - 0.66	0.40 - 0.94
2	1.48 - 1.78	0.66 - 1.28	0.94 - 1.00
3	1.98 - 2.40	1.48 - 1.78	1.00 - 1.02
4		1.98 - 2.40	1.02 - 1.12
5			1.12 - 1.16
6			1.16 - 1.28
7			1.48 - 1.78
8			1.98 - 2.40

EigenVector#	4	5	6
BAND			
1	0.40 - 0.92	0.40 - 0.70	0.40 - 0.44
2	0.92 - 1.26	0.70 - 0.92	0.44 - 0.50
3	1.26 - 1.28	0.92 - 0.96	0.50 - 0.52
4	1.48 - 1.78	0.96 - 1.06	0.52 - 0.66
5	1.98 - 2.40	1.06 - 1.28	0.66 - 0.84
6		1.48 - 1.78	0.84 - 0.92
7		1.98 - 2.40	0.92 - 0.94
8			0.94 - 1.00
9			1.00 - 1.04
10			1.04 - 1.12
11			1.12 - 1.28
12			1.48 - 1.64
13			1.64 - 1.78
14			1.98 - 2.20
15			2.20 - 2.40

An algorithm is developed to automatically choose the linearly independent bands from the first 6 eigenfunctions. Table 3.5 shows the result. Basically, this algorithm checks the rank of the matrix consisting of the bands

derived in Table 3.4. First, the linearly dependent bands in Table 3.4 are ranked from the widest to the narrowest. Then, starting from the widest band, this algorithm checks the matrix rank. If the rank is less than the total number of the band features, the band features in the matrix are linearly dependent, the widest linearly dependent band in the matrix is then eliminated from the set. On the other hand, if the rank is equal to the total number of the band features, increase the matrix rank by one and test the next widest band.

The procedure used in the above overlapping band feature selection algorithm can find the largest set of smallest bands that are linearly independent . This procedure can be summarized as follows :

(1)   Find the band edges of each individual eigenfunction

(2)   Rank these linearly dependent bands from the widest to the narrowest, then set rank $n = 1$

(3)   Starting from the widest band, check the rank of the feature matrix

(4)   If the rank is less than the total number of the bands, eliminate the widest linearly dependent band in the matrix, then go to step (3) to test the next widest band;

(5)   If the rank is equal to the total number of the bands, increase n by 1, then go to step (3) to test the next widest band

(6)   Set up the final feature set

Table 3.5 Linearly Independent Bands Found by Overlapping Band Feature Selection Algorithm for Data Set K2

Band	wavelength (µm)
1	0.70 - 0.92
2	1.98 - 2.20
3	2.20 - 2.40
4	0.66 - 0.84
5	1.48 - 1.64
6	0.52 - 0.66
7	1.64 - 1.78
8	1.16 - 1.28
9	0.96 - 1.06
10	1.04 - 1.12
11	0.94 - 1.00
12	0.44 - 0.50
13	1.12 - 1.16
14	0.92 - 0.96
15	0.40 - 0.44
16	1.00 - 1.04
17	1.00 - 1.02
18	1.26 - 1.28
19	0.50 - 0.52
20	0.92 - 0.94

3.5 Experimental System

In order to process the data in a digital computer, the spectral reflectance function $X(\lambda)$, the weight function $W(\lambda)$, the optimal basis function $\Phi_i(\lambda)$ and the sequence of the optimal basis functions $\Phi(\lambda)$ are represented by their discrete approximations, vector X, diagonal matrix W, basis vector Φ_i and the matrix Φ respectively.

An experimental software system has been set up to test the four approaches developed in the previous sections. This system has been

implemented on IBM 3083 computer. A collection of field data consisting of spectral sample functions on three dates from Williams County, ND, and three dates from Finney County, KS, was available from the field measurement library at Purdue/LARS. The spectral functions were sampled at 0.02 μm over the range 0.4 to 2.4 μm, therefore, the dimensionality is 100.

The optimal features are found numerically by estimating the covariance matrix from the sample functions. Maximum likelihood estimates of the mean and covariance matrix are given [34] by

$$M_x = E(X) \approx \overline{X} = \frac{1}{N_s} \sum_{i=1}^{N_s} X_i \qquad (3.1)$$

and

$$K_x = \frac{1}{N_s - 1} \sum_{i=1}^{N_s} (X_i - \overline{X})(X_i - \overline{X})^T \qquad (3.2)$$

where N_s is the number of the sample functions and X_i is the ith sample vector. The covariance matrix is then used to solve the discrete form of the generalized Karhunen Loeve Equation [14,15] :

$$K_x W \Phi = \Phi \Gamma \qquad (3.3)$$

where the Φ, Γ and W are the eigenvectors, eigenvalues and the weight matrix, respectively. The solutions of the equation are the optimal features.

In order to find appropriate non-overlapping bands used in feature design, the non-overlapping band feature selection algorithm is applied to the

average of the first few eigenvectors. Three cases were studied, tests using the first 6, 12 or 24 eigenvectors in the algorithm. For the illustrative example shown in section 3.1, the second case is considered.

For overlapping band features, the infinite clipping procedure is applied to each individual eigenfunction. In this preliminary test the first 6 eigenfunctions from each of the 6 data sets are used. The locations of the important spectral bands are then extracted. After applying the overlapping band feature selection algorithm to the spectral bands derived above, the desired linearly independent (L.I.) band features are found.

The bands found by the above two algorithms, the Walsh functions or the infinite clipped optimal features developed from the structure similarity property are then used as spectral features to perform the linear transformation on the data sets.

$$y_i = \Phi_i^T W X \tag{3.4}$$

In order to test the spectral features thus determined, the probability of correct classification is estimated using them. To do so, the class-conditional statistics are first computed using the transformed data. An algorithm based on the maximum likelihood estimator [34] is then applied, where the class conditional statistics are assumed to be multivariate Gaussian.

3.6 Preliminary Results

After applying the N.O.L. band feature selection algorithm to the average of the first 6, 12 or 24 eigenvectors of the six test data sets, the band edges are found. Table 3.6 shows the results for the data set K2 for three different number of eigenvectors. These three feature sets are named as proposed sensor C1, C2 and C3 respectively. For brevity, they are denoted PC1, PC2 and PC3. On the other hand, the O.L. band feature selection algorithm is applied to the first 6 eigenfunctions, the result of the first 16 linearly independent bands is shown in Table 3.7 for data set K2.

Furthermore, the probabilities of correct classification using Landsat (LS) MSS bands, Thematic Mapper (TM) bands and the two sensors proposed in Wiersma's work (PA and PB) [14,15] are also computed here. Table 3.8 shows the band edges associated with each sensor [15].

Table 3.6 Bands Found by Non-Overlapping Band Feature Selection Algorithm for Data Set K2

Band	PC1	PC2	PC3
1	0.40 - 0.68	0.40 - 0.68	0.40 - 0.66
2	0.68 - 0.84	0.68 - 0.90	0.66 - 0.80
3	0.84 - 0.90	0.90 - 0.92	0.80 - 0.88
4	0.90 - 0.96	0.92 - 0.94	0.88 - 0.94
5	0.96 - 1.00	0.94 - 1.00	0.94 - 1.00
6	1.00 - 1.06	1.00 - 1.06	1.00 - 1.04
7	1.06 - 1.12	1.06 - 1.12	1.04 - 1.16
8	1.12 - 1.28	1.12 - 1.26	1.16 - 1.26
9	1.48 - 1.74	1.26 - 1.28	1.26 - 1.28
10	1.74 - 1.78	1.48 - 1.78	1.48 - 1.54
11	1.98 - 2.40	1.98 - 2.40	1.54 - 1.64
12			1.64 - 1.74
13			1.74 - 1.78
14			1.98 - 2.20
15			2.20 - 2.26
16			2.26 - 2.40

**Table 3.7   Bands Found by Overlapping Band Feature
Selection Algorithm for Data Set K2**

Band	wavelength (μm)
1	0.70 - 0.92
2	1.98 - 2.20
3	2.20 - 2.40
4	0.66 - 0.84
5	1.48 - 1.64
6	0.52 - 0.66
7	1.64 - 1.78
8	1.16 - 1.28
9	0.96 - 1.06
10	1.04 - 1.12
11	0.94 - 1.00
12	0.44 - 0.50
13	1.12 - 1.16
14	0.92 - 0.96
15	0.40 - 0.44
16	1.00 - 1.04

Figures 3.5 to 3.10 are the classification performance comparisons of the optimal functions (Optimal), Walsh functions (Walsh) and the infinite clipped optimal functions (Clipped) for the 6 data sets.   Figure 3.11 to 16 are the comparisons of the LS, TM, Wiersma's proposed sensor PA, non-overlapping band features (NOL) derived from the first 24 eigenfunctions (i.e., PC3), overlapping band features (OL), Walsh functions, infinite clipped optimal functions and optimal functions for the 6 preliminary test data sets.   From the implementation point of view, since there are only two values (+1, -1) for the Walsh functions and three values (+1, -1, 0) for the infinite clipped optimal functions, it can be concluded from Figures 3.5 to 3.16 that representing the optimal features using their infinite clipping versions or using the first 16 Walsh functions produces the more practical features used for classification which

provide a classification accuracy quite near that of optimal features. The classification performances estimated for the above sensors are shown in Table 3.9, where PC1, PC2 and PC3 represent the sensors derived from N.O.L. band feature selection algorithm using the first 6, 12 and 24 eigenvectors as their input respectively; Optimal, Walsh and Clipped stand for the sensors using the first 16 optimal functions, the first 16 Walsh functions and the first 16 infinite clipped optimal functions as spectral features respectively.

Table 3.8 Band Edges of Landsat MSS, TM, PA and PB Sensors

Band	LS	TM	PA	PB
1	0.50-0.60	0.45-0.52	0.42-0.54	0.42-0.66
2	0.60-0.70	0.52-.060	0.56-0.66	0.68-0.70
3	0.70-0.80	0.63-0.69	0.68-0.70	0.72-0.92
4	0.80-1.10	0.76-0.90	0.72-0.90	0.94-1.04
5		1.55-1.75	0.92-1.00	1.06-1.10
6		2.08-2.35	1.02-1.30	1.12-1.30
7			1.52-1.74	1.52-1.74
8			1.96-2.40	1.96-2.40

Table 3.9 Probability of Correct Classification for 6 Data Sets

SENSOR	K1	K2	K3	N1	N2	N3
LS	0.90	0.78	0.85	0.77	0.83	0.96
TM	0.92	0.79	0.93	0.89	0.95	0.99
PA	0.94	0.86	0.95	0.92	0.96	0.99
PB	0.94	0.85	0.94	0.89	0.96	0.96
PC1	0.94	0.87	0.96	0.92	0.97	0.99
PC2	0.96	0.88	0.97	0.94	0.97	0.99
PC3 (NOL)	0.96	0.94	0.98	0.96	0.98	0.99
OL	0.97	0.94	0.98	0.97	0.99	0.99
Walsh	0.98	0.95	0.98	0.95	0.98	0.99
Clipped	0.98	0.97	0.99	0.97	0.99	0.99
Optimal	0.98	0.97	0.98	0.97	0.99	0.99
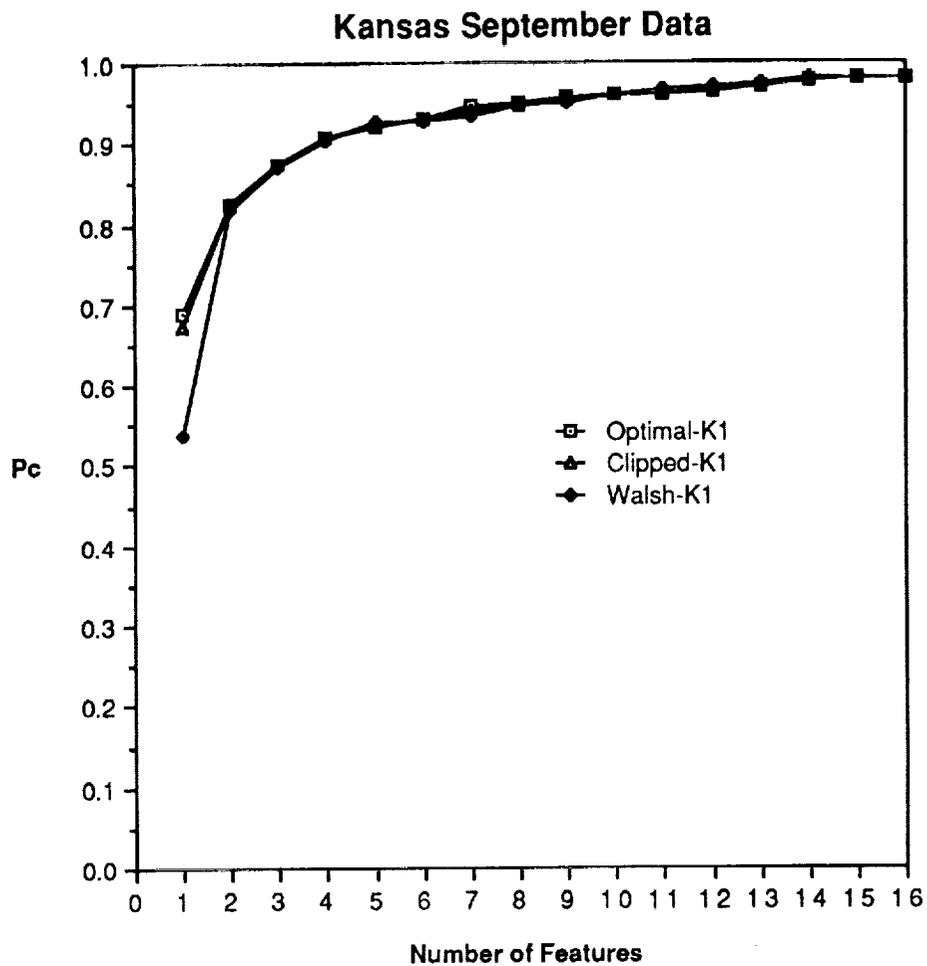
Figure 3.5 **Performance Comparison of Optimal, Infinite Clipped Optimal and Walsh Functions for Data Set K1**

Kansas May Data
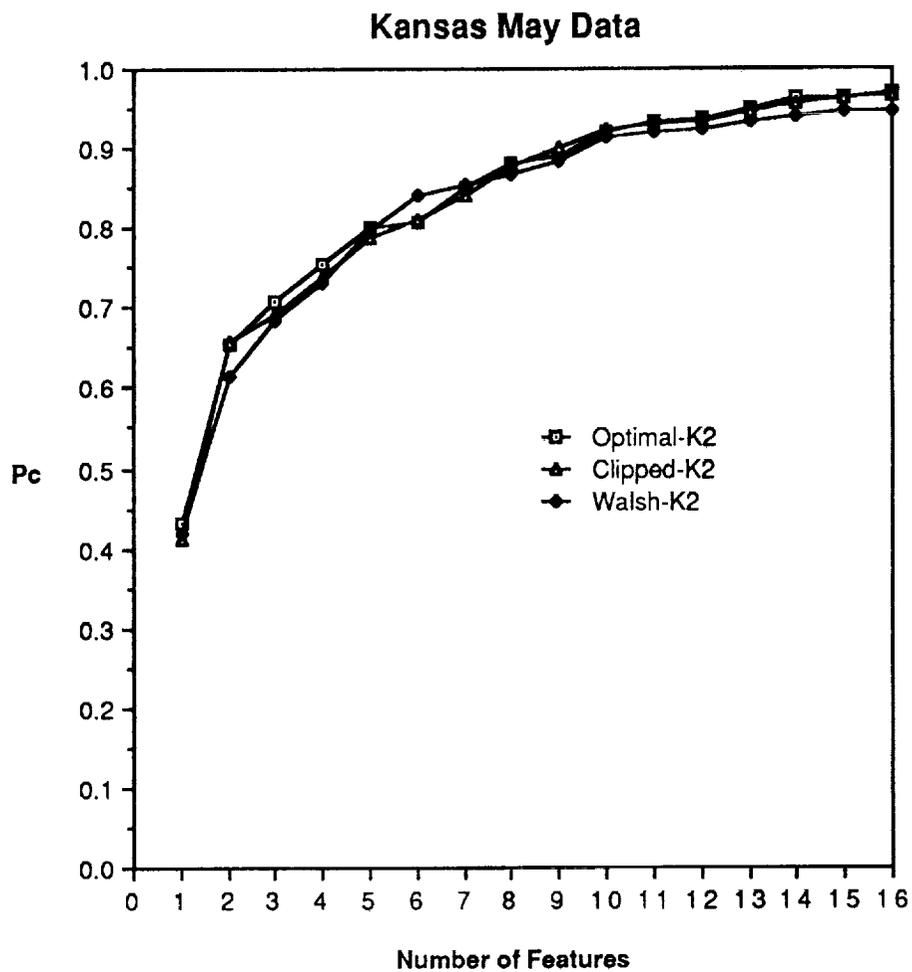
Figure 3.6   Performance Comparison of Optimal, Infinite Clipped Optimal and Walsh Functions for Data Set K2

Kansas June Data
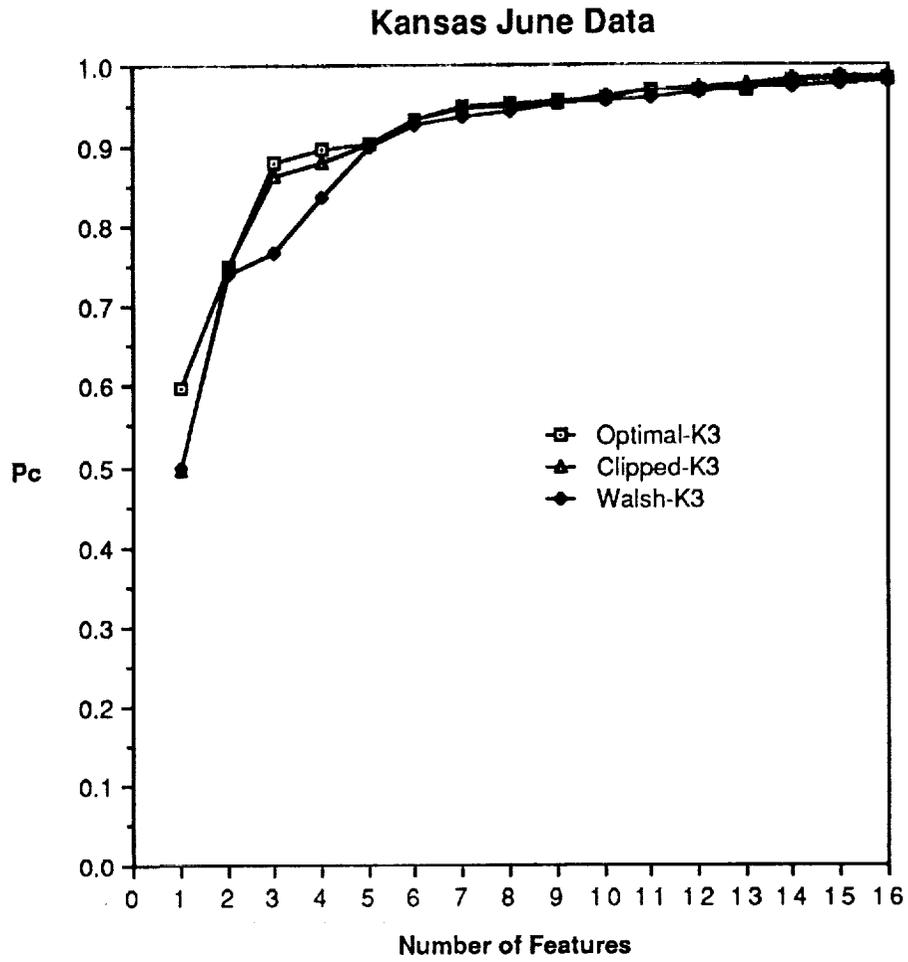
Figure 3.7   **Performance Comparison of Optimal, Infinite Clipped Optimal and Walsh Functions for Data Set K3**

North Dakota May Data

Legend:
- Optimal-N1
- Clipped-N1
- Walsh-N1

Y-axis: Pc
X-axis: Number of Features
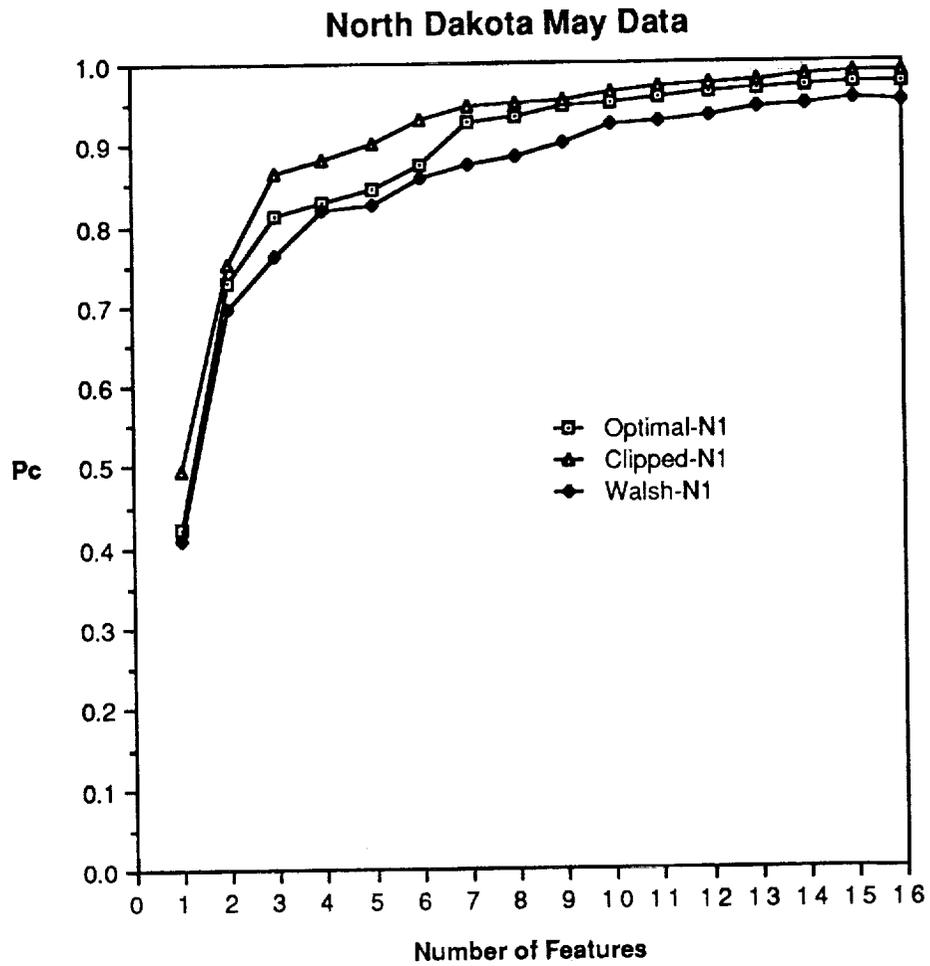
Figure 3.8 Performance Comparison of Optimal, Infinite Clipped Optimal and Walsh Functions for Data Set N1

North Dakota June Data
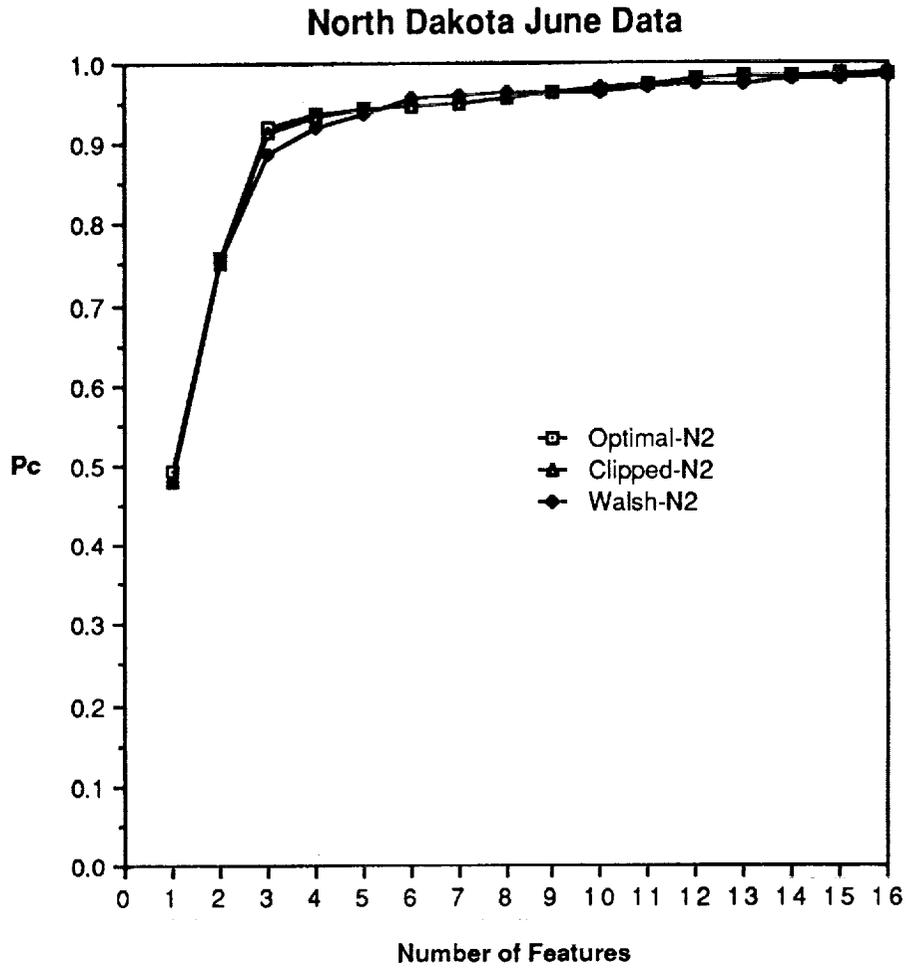
Figure 3.9 **Performance Comparison of Optimal, Infinite Clipped Optimal and Walsh Functions for Data Set N2**
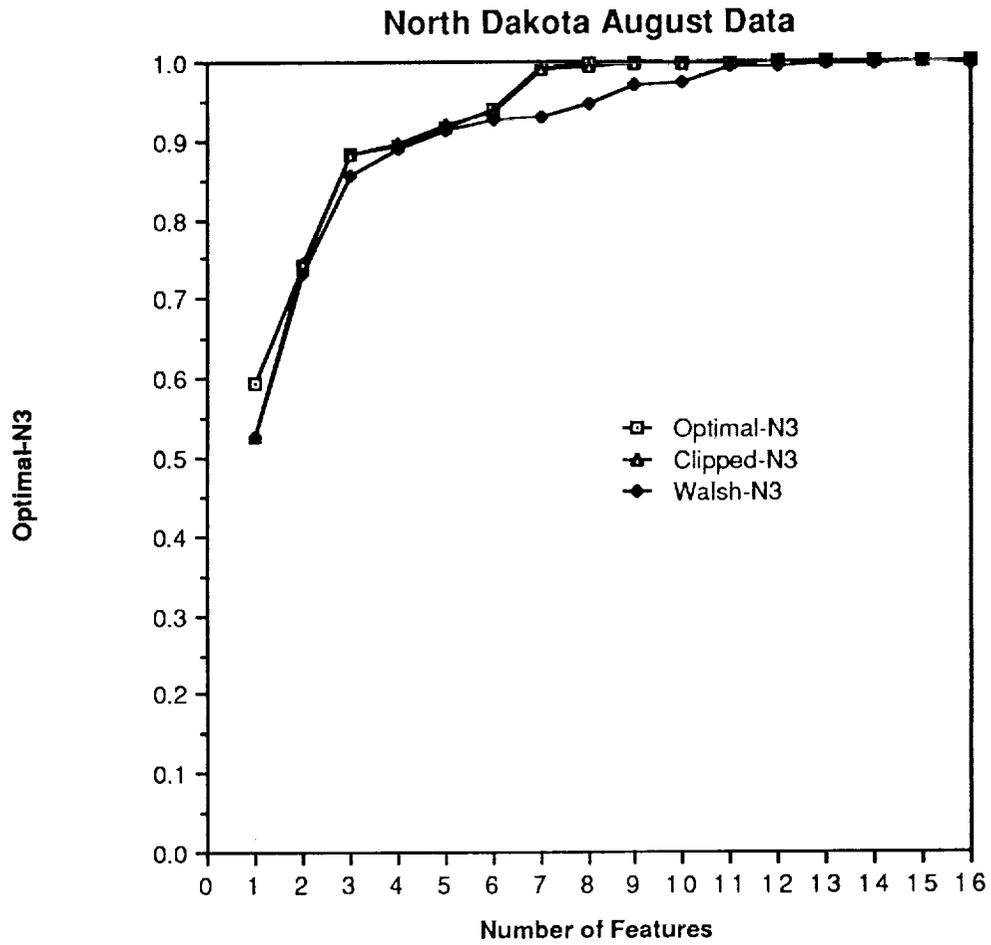
Figure 3.10 Performance Comparison of Optimal, Infinite Clipped Optimal and Walsh Functions for Data Set N3

Kansas September Data
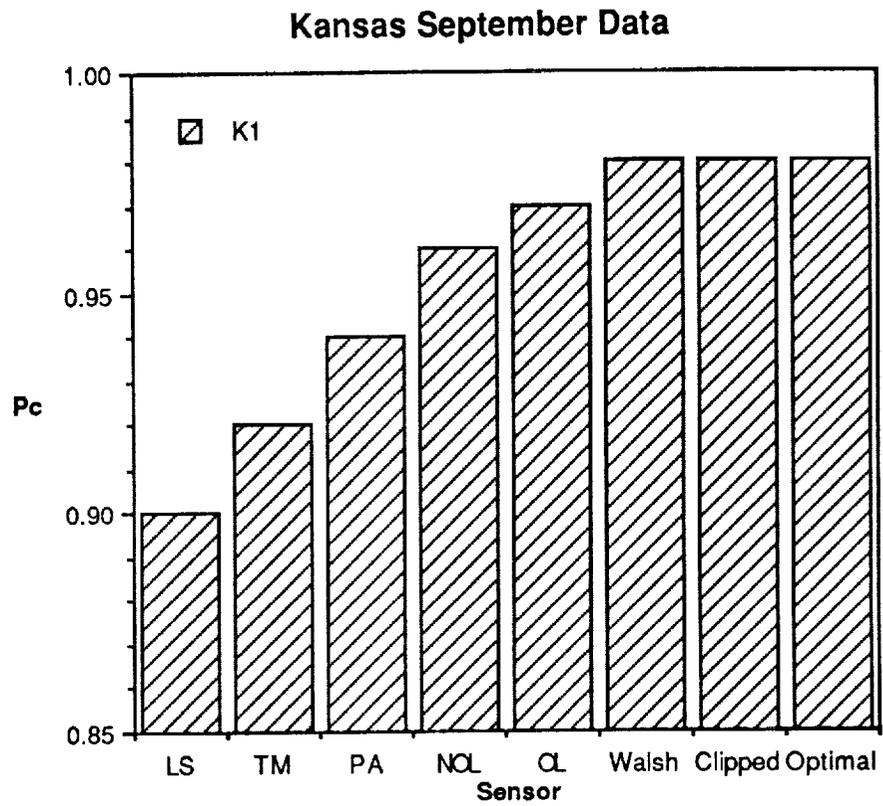
Figure 3.11 Performance Comparison for Data Set K1

Kansas May Data
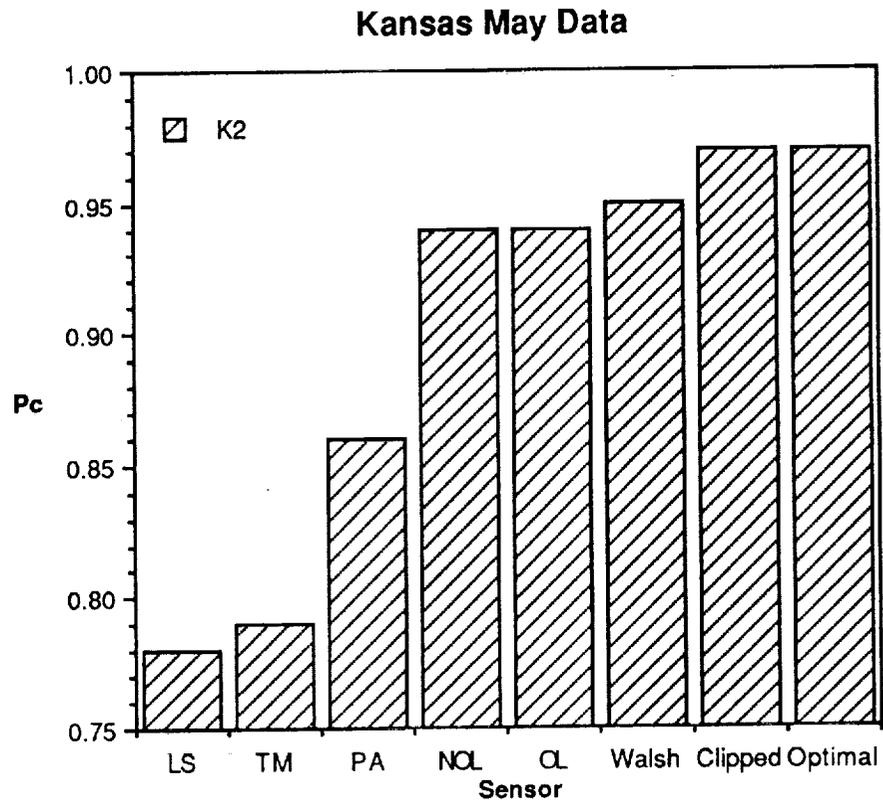
Figure 3.12 Performance Comparison for Data Set K2
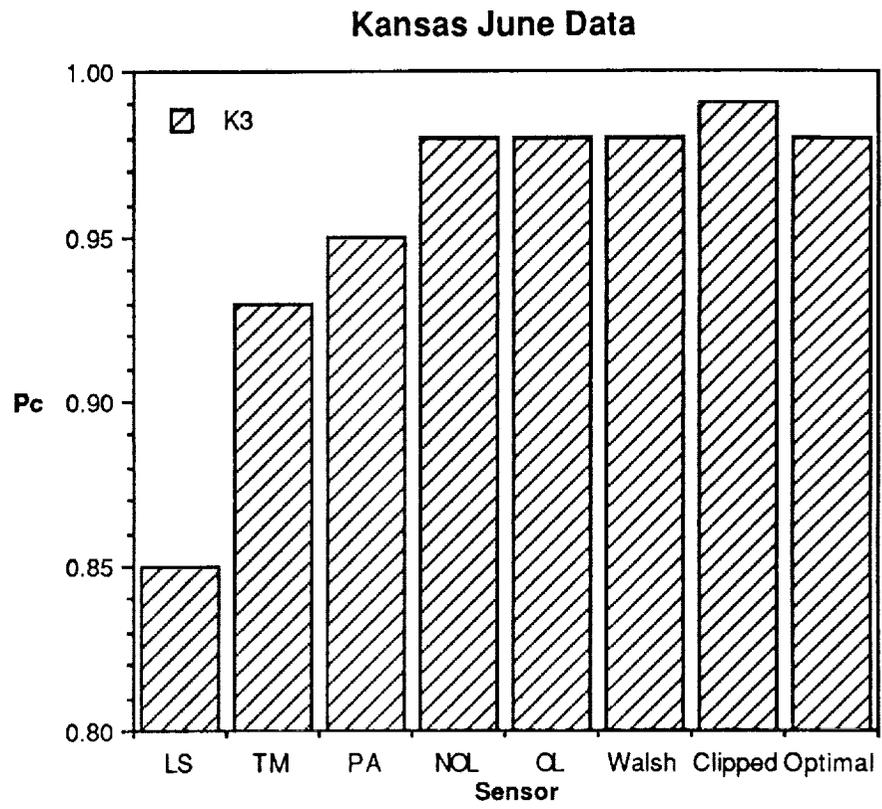
Figure 3.13 Performance Comparison for Data Set K3
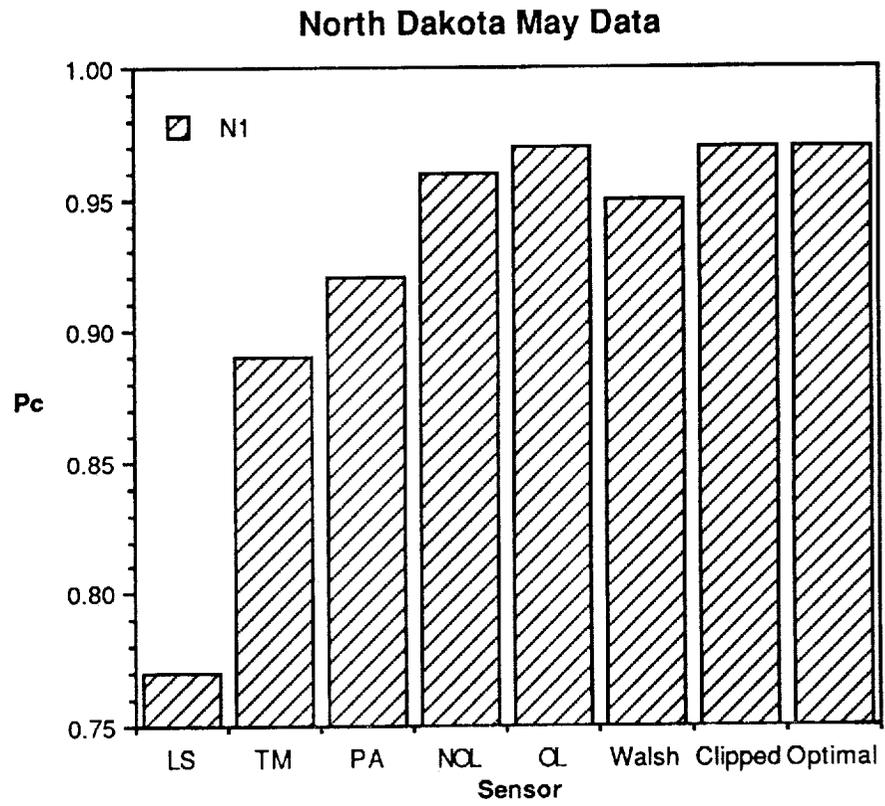
Figure 3.14 Performance Comparison for Data Set N1
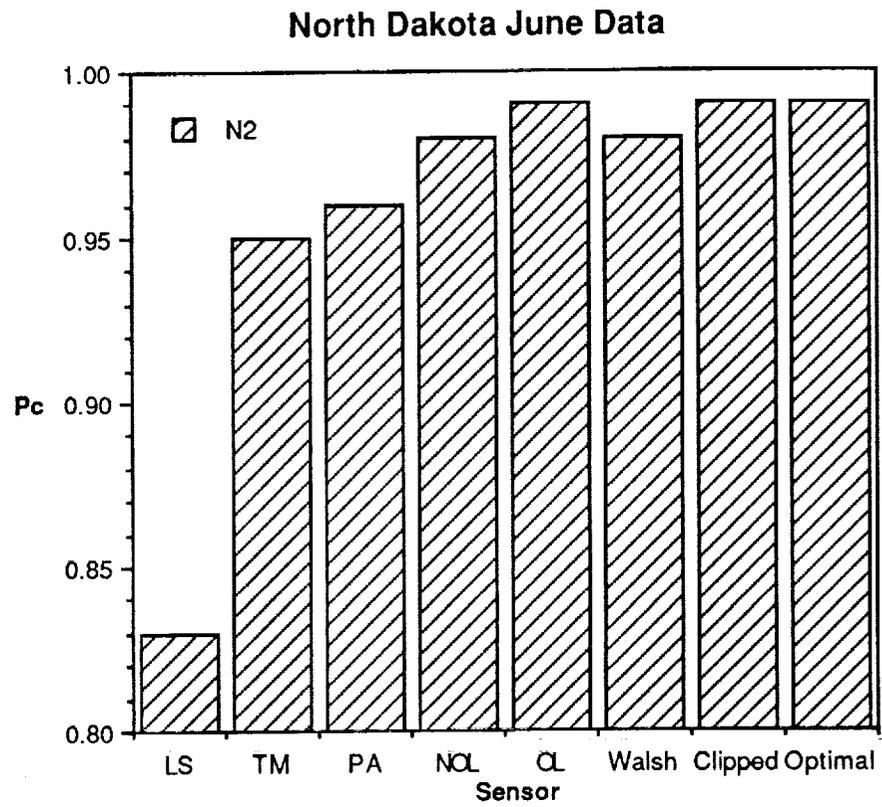
Figure 3.15 Performance Comparison for Data Set N2
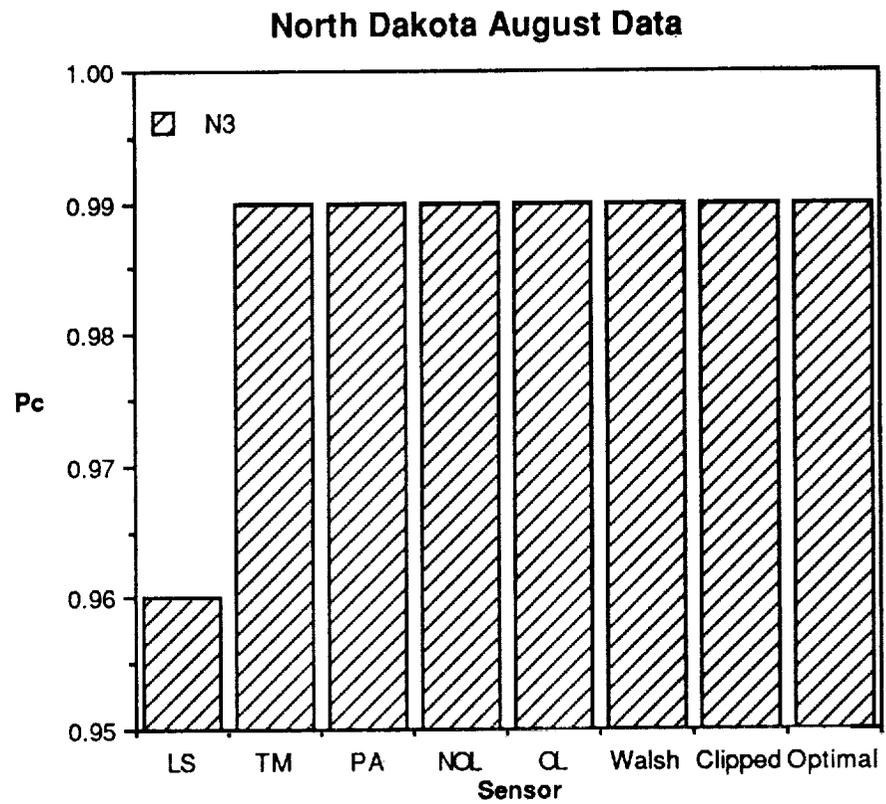
Figure 3.16 Performance Comparison for Data Set N3

3.7 Selection of the Best On-Board Preprocessing Scheme

From Table 3.9 and Figures 3.5 to 3.16, it is seen that the four approaches developed in this research, two based on the " shape " of the optimal features and the other two from their "structure" similarity with the optimal functions, are feasible ways for feature design.

The fundamental objective of this research is to develop an objective and practical spectral feature design technique for high dimensional multispectral data. There are two important factors, simplicity and effectiveness, which must be considered in this respect.

First of all, from simplicity point of view, the overlapping band feature selection algorithm is harder to perform than the other three because of the existence of linear dependence problem. In order to find appropriate overlapping band features, we have to check the rank of the matrix for each newly selected band. This procedure needs more time than the other three approaches. However, its classification performance [referring to Figure 3.11 to 3.16] does not indicate much advantage over the other three, especially the infinite clipped optimal function approach.

For example, Figure 3.11 and 3.12 show that for Kansas September and Kansas May data the performances of the overlapping band feature selection algorithm are the 3rd best among the four techniques. The infinite clipped optimal function approach and the Walsh function approach have better performances than that of the overlapping band feature selection algorithm. Figure 3.13 to 3.16 indicate that the performances of the overlapping band feature selection algorithm are never better than those of the infinite clipped

optimal function approach. Therefore, from simplicity point of view, the overlapping band feature selection algorithm would not be used in this thesis as the best technique for the final data preprocessing system.

On the other hand, from effectiveness point of view, referring to Table 3.9 and Figure 3.5 to 3.16 again, it is shown that the infinite clipped optimal transform has better performance than the Walsh transform and the non-overlapping band feature selection algorithm.

For instance, Figure 3.5 to 3.10 indicate that the infinite clipped optimal features have better classification accuracy than the Walsh features for all the six preliminary test data sets in Kansas and North Dakota. Figure 3.11 to 3.16 show that the infinite clipped optimal features perform better than the non-overlapping band features for all the 6 test data sets except for North Dakota August data (Figure 3.16) where these two techniques have the same performance.

Therefore, from simplicity and effectiveness point of view, the infinite clipped optimal transform is chosen to be the best scheme in the data preprocessing stage of the spectral feature design system.

The processing up to this point, consisting of the optimal features calculation, the infinite clipping, and the data transform is based solely upon the ensemble statistics of the field data. Additional a priori knowledge that might be used to improve the performance is the class statistics of the scene. The objective is then to find the best features under the criterion of maximal class separability.

3.8 Canonical Analysis and Ground Station Data Processing.

Canonical Analysis is a technique that can be used to find the optimal features under a maximal separability criterion [36-41]. Unlike principal component analysis, which is based on the global covariance matrix of the full data set, canonical analysis utilizes the class structure of the data. The advantage of canonical analysis is its ordering property on the separability measure. By using the features derived from canonical analysis to further process the received transformed data, the classification performance should, therefore, be improved.

Let M_i and S_i be the ith class mean vector and covariance matrix of a data set with L classes. In canonical analysis one first finds the within-class scatter and the among-class scatter matrices S_w and S_a respectively :

$$S_w = \sum_{i=1}^{L} \frac{(N_i - 1)}{N_s} * S_i \qquad (3.5)$$

where N_i is the number of samples of the ith class data and N_s is the total number of samples of the ensemble. And,

$$S_a = \frac{1}{L} \sum_{i=1}^{L} (M_i - M_0)(M_i - M_0)^T \qquad (3.6)$$

where M_0 is the global mean, given by

$$M_0 = \sum_{i=1}^{L} \frac{N_i}{N_s} * M_i \qquad (3.7)$$

The within class scatter matrix, S_w, is an average quantity that describes how closely the samples are distributed around their class means while the among class scatter matrix, S_a, is a quantity measuring the average degree of closeness between the ensemble mean and each class mean. The optimally separable feature is a feature such that S_w is minimized and S_a is maximized after the transformation. Define a quantity r and let the desired feature be vector **d**. Then the objective is to find the r and **d** that result in maximal class separability. That is,

$$r = \frac{d^T S_a d}{d^T S_w d}$$ (3.8)

must be maximized. The ratio of variances in the new space is maximized by the selection of feature **d** if,

$$\frac{\partial r}{\partial d} = 0$$ (3.9)

The above equation can be reduced to

$$(S_a - r * S_w) * d = 0$$ (3.10)

which is called a generalized eigenvalue equation and must be solved now for the unknown r and **d**. The first canonical axis will be in the direction of **d**, and r will give the associated ratio of among-class to within-class variance for that axis.

The development to this stage is usually referred to as discriminant analysis. One more step is included in the case of canonical analysis where the derived canonical features are normalized with respect to the within class scatter matrix. That is,

$$D^T * S_w * D = I \qquad (3.11)$$

where D is the matrix of canonical features d. This says that the within class scatter matrix after the transformation must be the identity matrix. In other words, after transformation, the classes should appear spherical.

CHAPTER IV

RESULTS AND DISCUSSIONS

In the previous chapter, we have introduced the four spectral feature design techniques developed in the course of this research. Six preliminary test data sets in Kansas and North Dakota were used to test the schemes. From a simplicity and effectiveness point of view, the infinite clipped optimal transform is chosen as the better means for data preprocessing. Furthermore, canonical analysis is applied to the above received transformed data on the ground station to achieve the maximal class separability. In this chapter, both the vegetation and the soil data will be used to find the classification performance for the final spectral feature design system. The spectral range for the vegetation data is from 0.4 μm to 2.4 μm with resolution 0.02 μm while the range for the soil data is from 0.45 μm to 2.45 μm with resolution 0.01 μm. Therefore the dimensionality for the vegetation data and the soil data is 100 and 200 respectively. The final results of these data will be presented in section 4.1 and 4.2. Moreover, due to the limited sample size of the data set to estimate the covariance matrix, different degree of Hughes phenomenon occurs in some of the one-day Kansas and North Dakota vegetation data sets as well as in all soil data sets. This effect will be discussed in section 4.3.

4.1 Vegetation Data

Four sets of multitemporal multispectral data collected in Kansas, North Dakota, Iowa and South Dakota are acquired to test the proposed spectral feature design system. Table 4.1 show the species, the dates on which the data were collected, and the total numbers of sample functions for each information class. In Table 4.1, the numbers appearing in the parentheses are the total numbers of sample functions collected for that class. Furthermore, W.Wheat and S.Wheat stand for winter wheat and spring wheat respectively.

Figure 4.1 to 4.6 show the probability of correct classification, Pc, using the optimal features, infinite clipped optimal features and features that are derived from infinite clipped optimal transform and canonical analysis for the six preliminary test data sets. These 6 data sets are part of the multitemporal data in Kansas and North Dakota (referring to Table 1.1 and Table 4.1). Each one of them consists of the sample functions collected on one single date and has 3 informational classes. The results indicate that using the first 16 infinite clipped versions of the optimal functions, 95% classification accuracy can be achieved.

Another important point is the occurrence of Hughes phenomenon [42,43] shown in Figure 4.1 to 4.4. It says that for data set K1, K2, K3 and N1, increasing the computational complexity [11] does not always increase the classification performance. For example, Figure 4.1 shows that canonical analysis improves the accuracy for the first 3 features, but it does not help beyond this complexity for data set K1. Figure 4.2 to 4.4 show that canonical analysis can only have better performance for the first 4 features for data sets K2, K3 and N1 respectively.

For data set N2 and N3, it is found in Figure 4.5 and 4.6 that Hughes phenomenon does not occur, and the classification performance using the features derived from infinite clipped optimal transform and canonical analysis is always better than those of the optimal features and the infinite clipped optimal features. It is also shown that only 2 features are needed to have about 94% and 99% classification accuracy for these 2 data sets respectively.

Figure 4.7 and 4.8 show the results for Kansas and North Dakota multi-temporal data. Each one has 9 information classes collected on 3 different dates from 1976 to 1977. The results indicate that canonical analysis improves the accuracy by about 15% to 25% for the first feature and about 1% for the first 16 features. Figure 4.9 is the results of Kansas and North Dakota combined data with 18 information classes. It is used to show the robustness property of this spectral feature design system. The results show that the technique is not overly sensitive for spatially and temporally combined data.

Figure 4.10 and 4.11 are the results for 25-class Iowa and 42-class South Dakota multi-temporal data. They are used to show the capability of this spectral feature design system for complex data sets. It can be seen that the system is very successful in this respect.

Table 4.1 : Vegetation Data Sets.
Numbers in the parenthesis are the total numbers of samples.

Kansas Vegetation Data Set : 9 classes

9/28/76	5/3/77	6/26/77
W.Wheat (141)	W.Wheat (658)	W.Wheat (677)
Summer Fallow (414)	Summer Fallow (211)	Summer Fallow (643)
Sorghum (277)	Unknown Class (682)	Sorghum (157)

North Dakota Vegetation Data Set : 9 classes

5/8/77	6/29/77	8/4/77
S.Wheat (664)	S.Wheat (787)	S.Wheat (931)
Summer Fallow (437)	Summer Fallow (291)	Summer Fallow (330)
Pasture (164)	Pasture (161)	Pasture(183)

Iowa Vegetation Data Set : 25 classes collected on 9 different dates of 1979;

5/15/79	5/23/79	6/11/79	6/29/79	7/16/79	7/17/79	8/30/79	10/25/79	11/2/79
Corn (514)	Corn (517)	Corn (621)	Corn (610)	Corn (437)	Corn (190)	Corn (650)	Corn (435)	Corn (393)
	Soybeans (36)	Soybeans (517)	Soybeans (485)	Soybeans (377)	Soybeans (172)	Soybeans (568)	Soybeans (417)	Soybeans (267)
Oats (41)	Oats (32)	Oats (45)	Oats (21)	Oats (22)	Oats (25)	Oats (42)	Oats (44)	

South Dakota Vegetation Data Set : 42 classes collected on 6 different dates of 1978 and 1979

9/21/78	10/26/78	6/1/79	6/21/79	7/25/79	8/11/79
Pasture (225)	Pasture (217)				
Alfalfa (61)	Alfalfa (51)			Alfalfa (45)	Alfalfa (42)
W.Wheat (292)	W.Wheat (393)				
S.Wheat (469)	S.Wheat(441)	S.Wheat(118)	S.Wheat(121)	S.Wheat (102)	S.Wheat (119)
Barley (82)	Barley (80)	Barley (43)	Barley (44)	Barley (66)	Barley (69)
Oats (182)	Oats (88)			Oats (89)	Oats (76)
IdleLand (63)					
Sorghum (103)	Sorghum (88)			Sorhgum (78)	Sorhgum (96)
Sunflower (39)	Sunflower (41)			Sunflower (53)	Sunflower (107)
Corn (39)	Corn (32)			Corn (147)	Corn (154)
	Millet (26)			Millet (39)	Millet (28)
				Safflower (24)	Safflower (19)

Kansas September Data
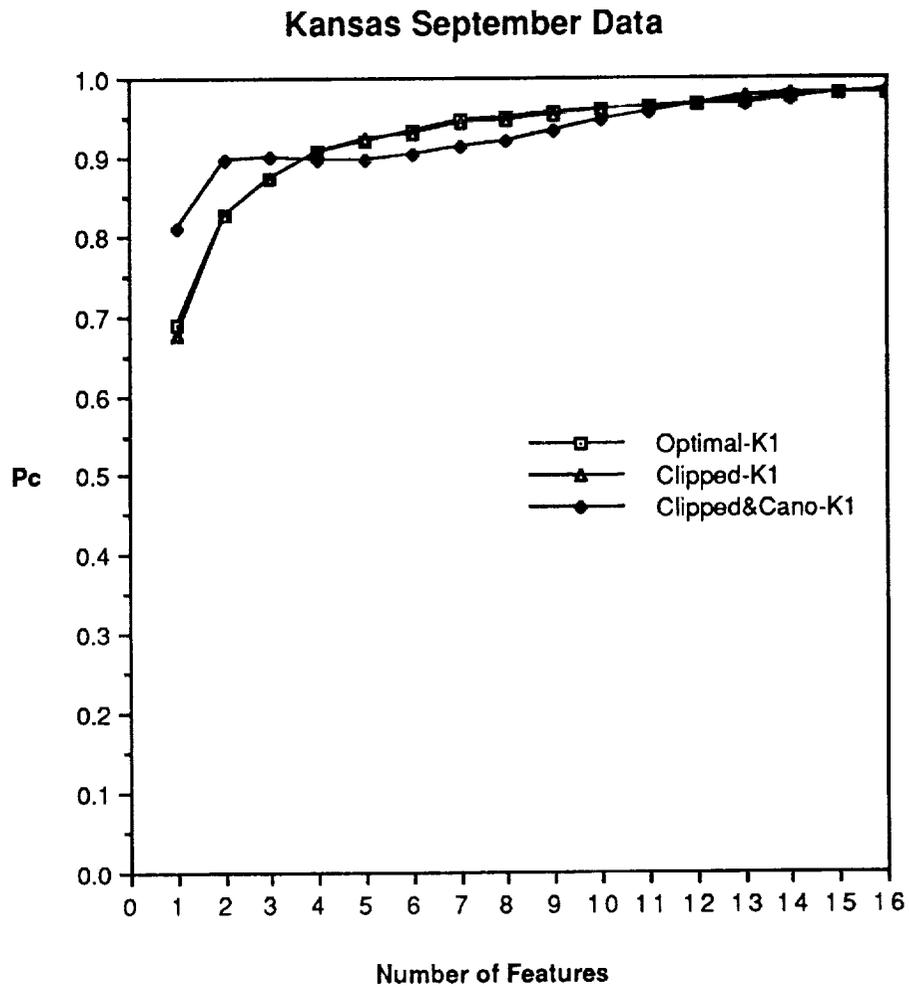
Figure 4.1 Classification Performance for Data Set K1

Kansas May Data
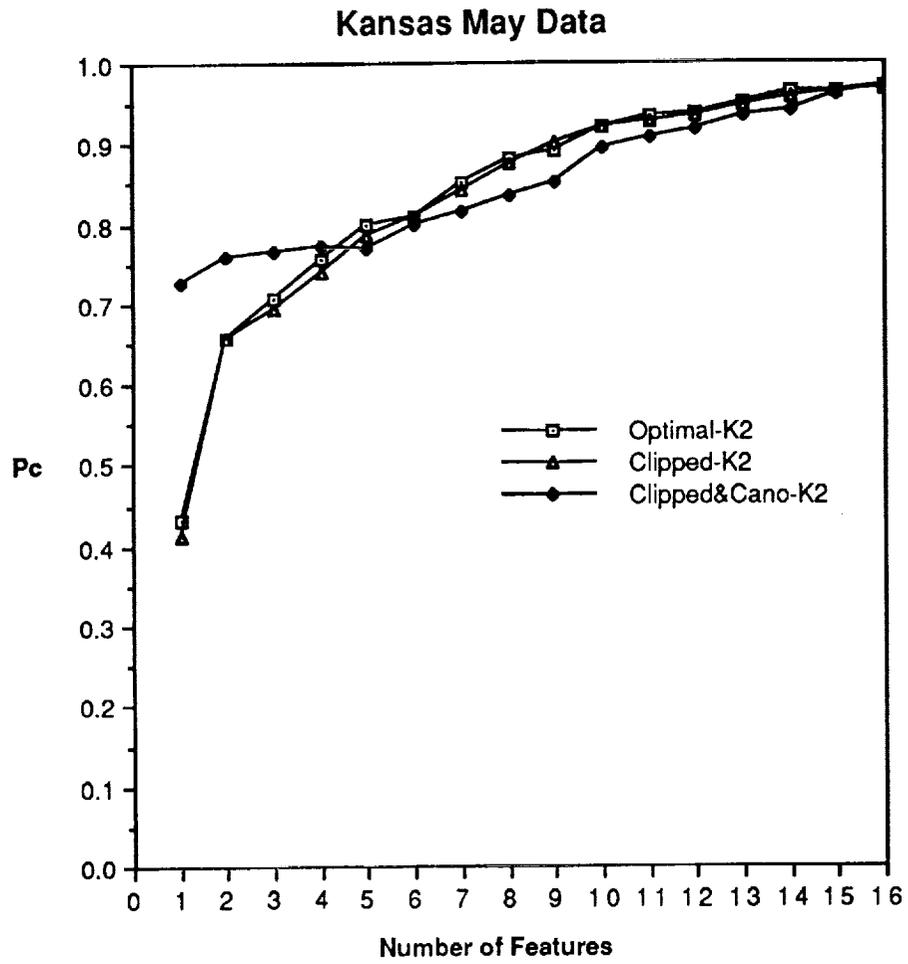
Figure 4.2 Classification Performance for Data Set K2
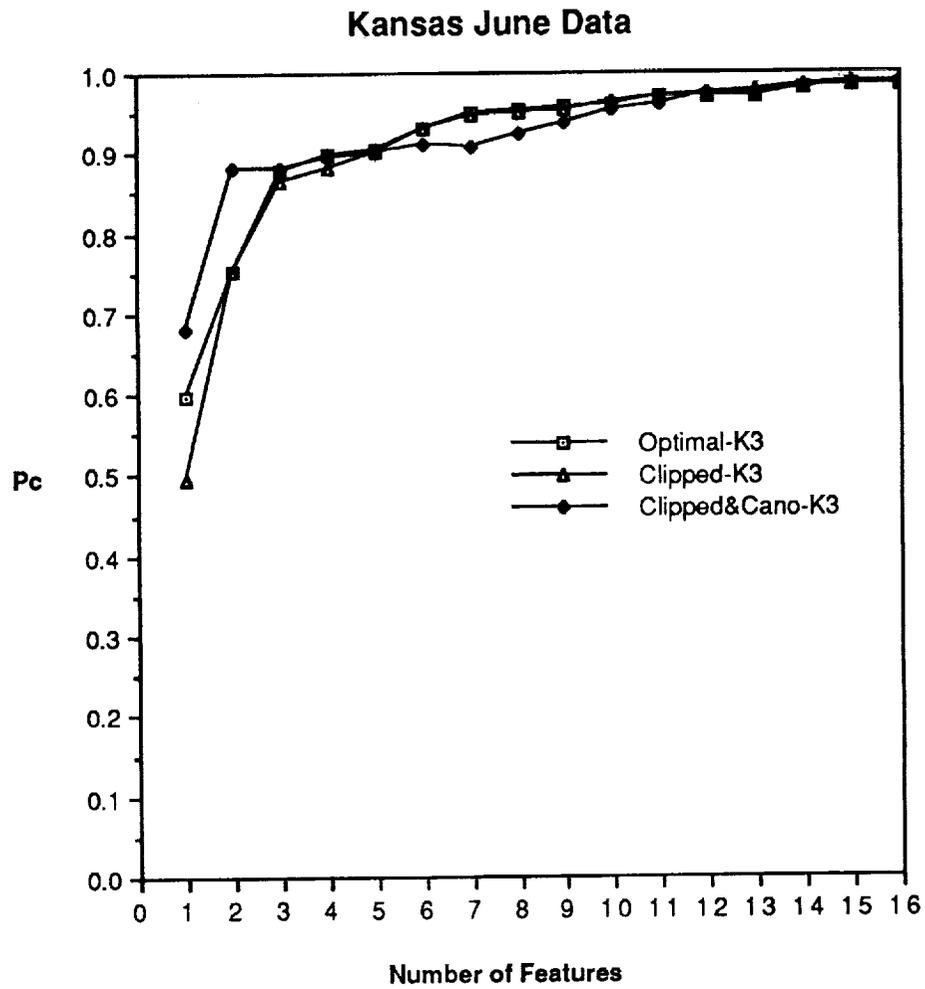
Figure 4.3   Classification Performance for Data Set K3
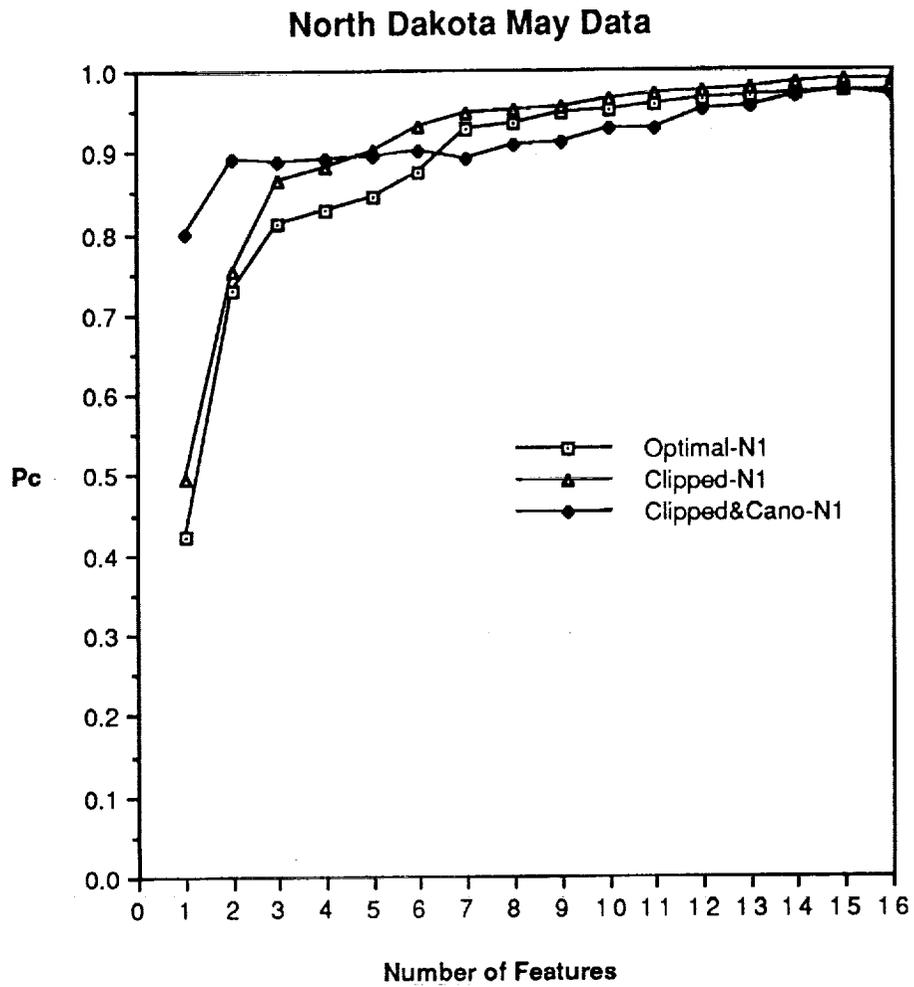
Figure 4.4 Classification Performance for Data Set N1

North Dakota June Data
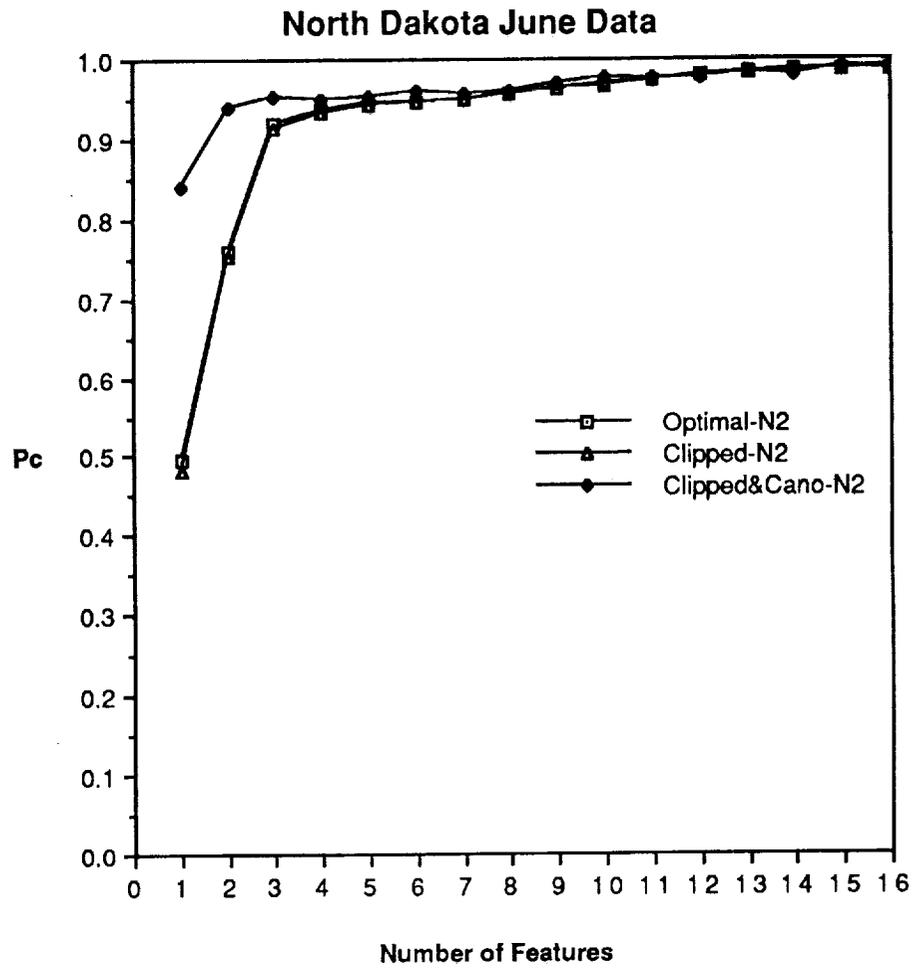
Figure 4.5   Classification Performance for Data Set N2

North Dakota August Data
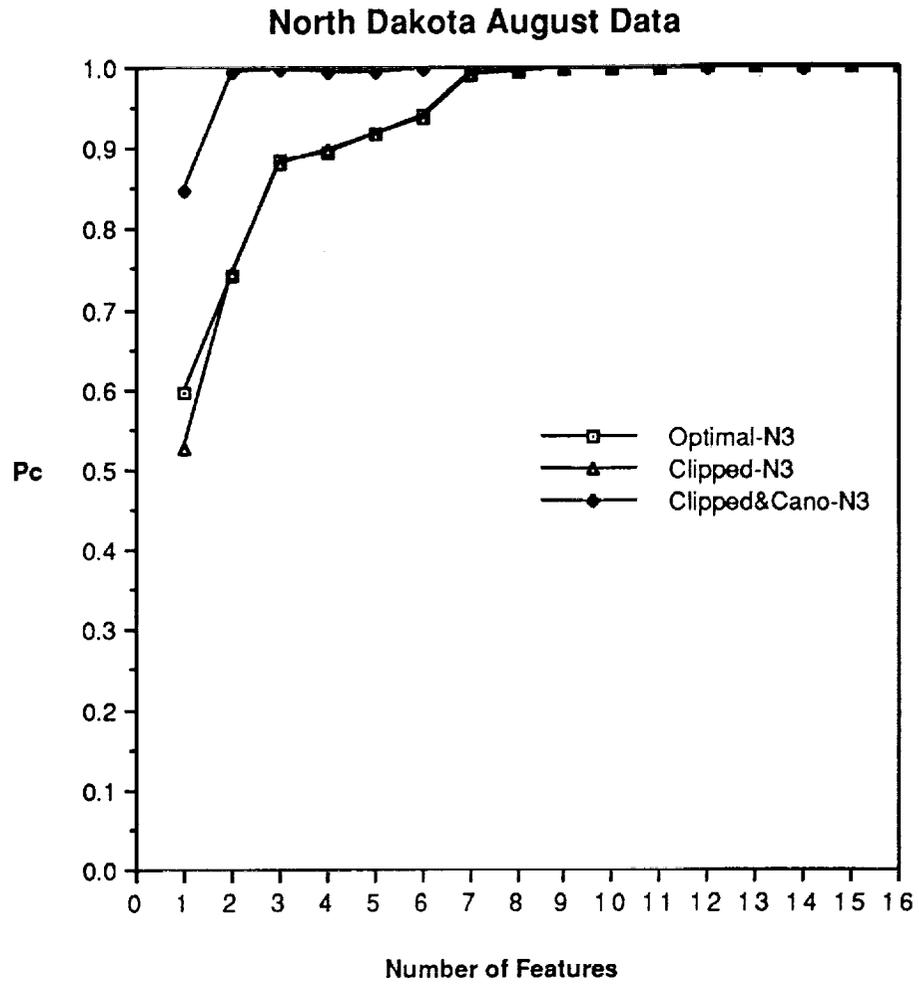
Figure 4.6   Classification Performance for Data Set N3

Kansas Vegetation Data
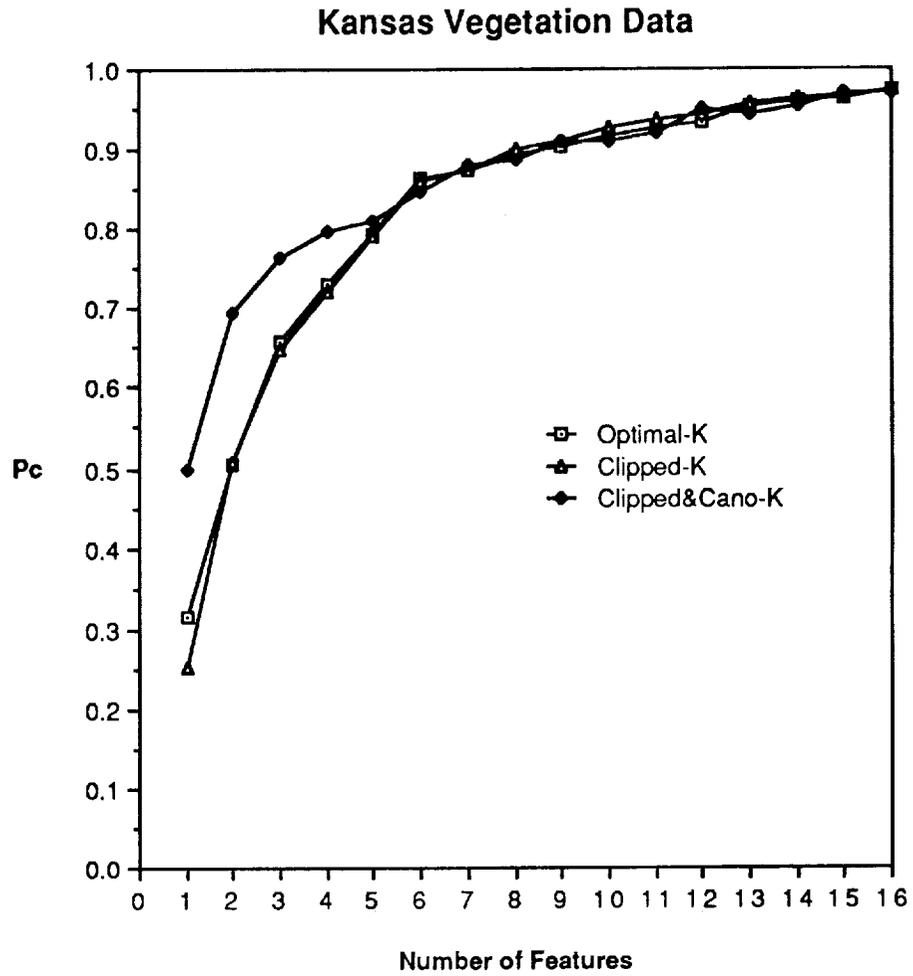
Figure 4.7   Classification Performance for Kansas
Multitemporal Data Set

North Dakota Vegetation Data
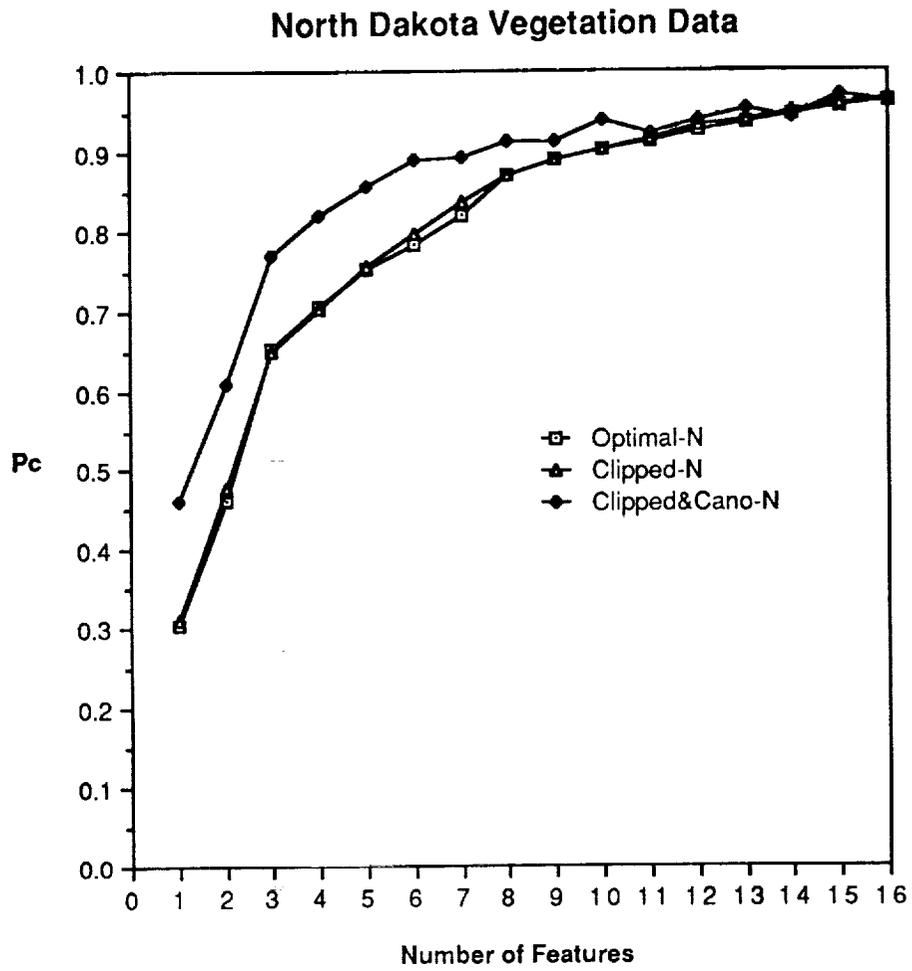
Figure 4.8 Classification Performance for N. Dakota Multitemporal Data Set
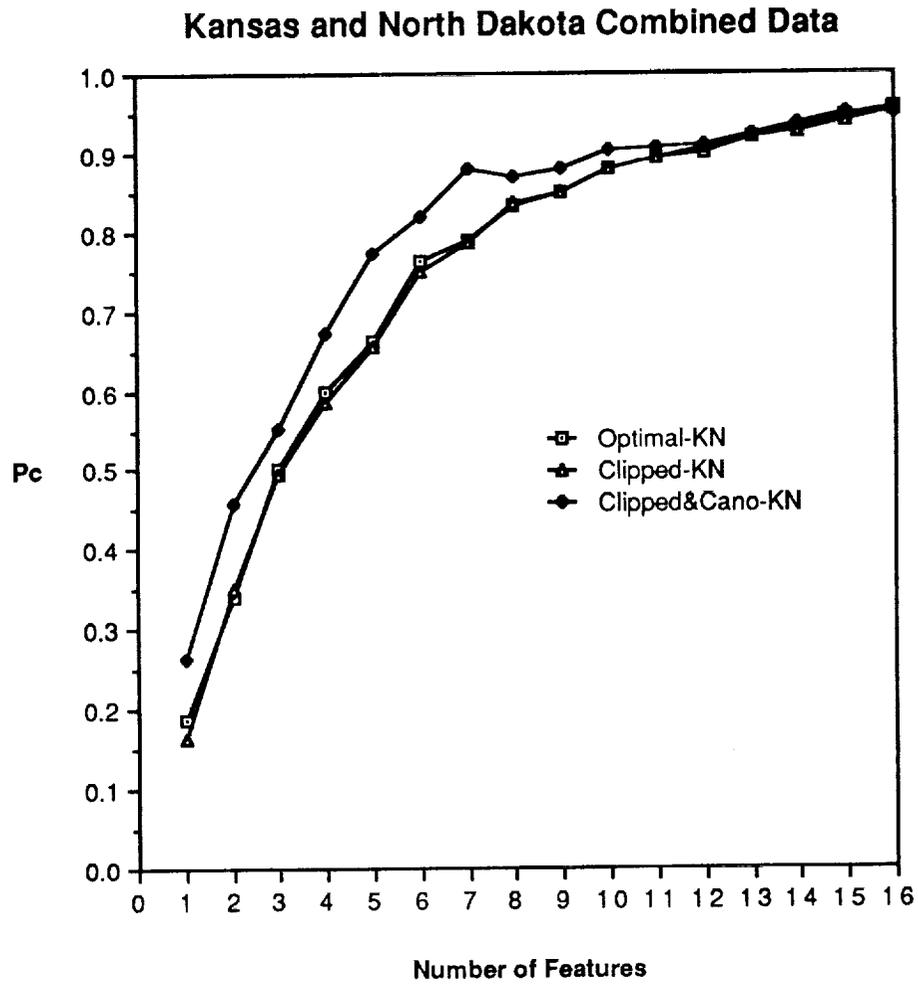
Figure 4.9   Classification Performance for
KS/ND Combined Data Set
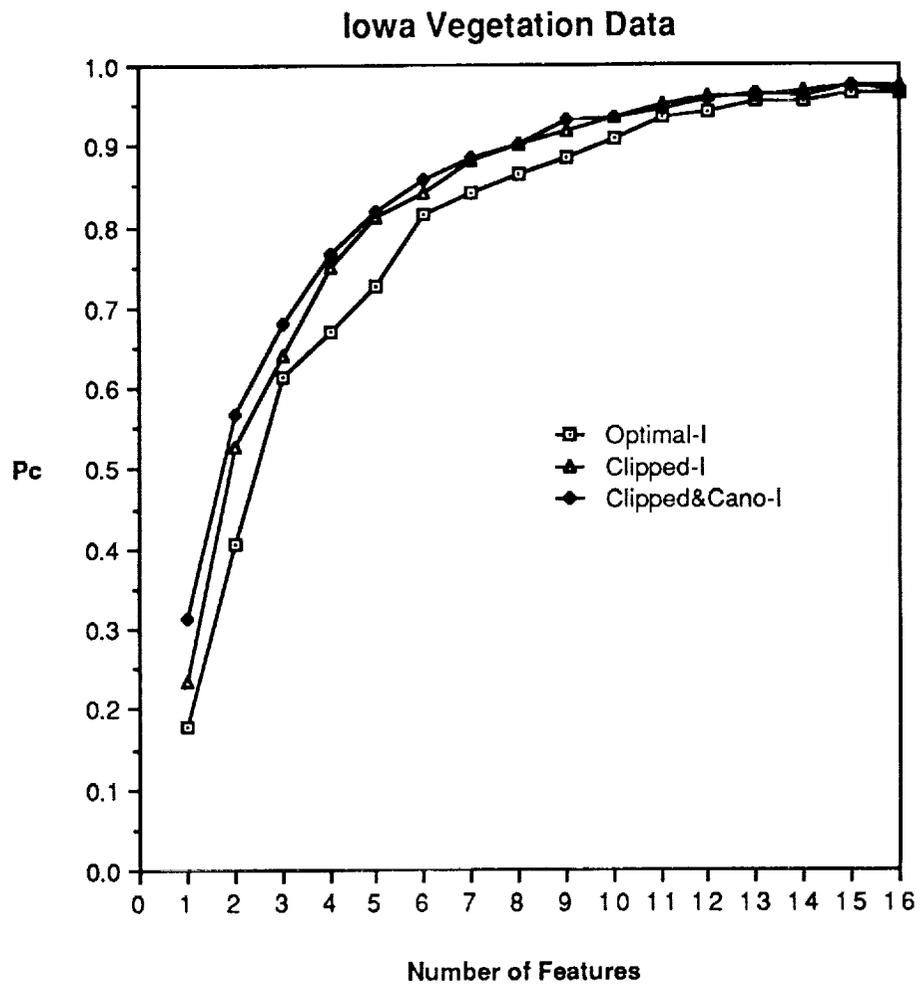
Figure 4.10   Classification Performance for Iowa
Multitemporal Data Set

South Dakota Vegetation Data
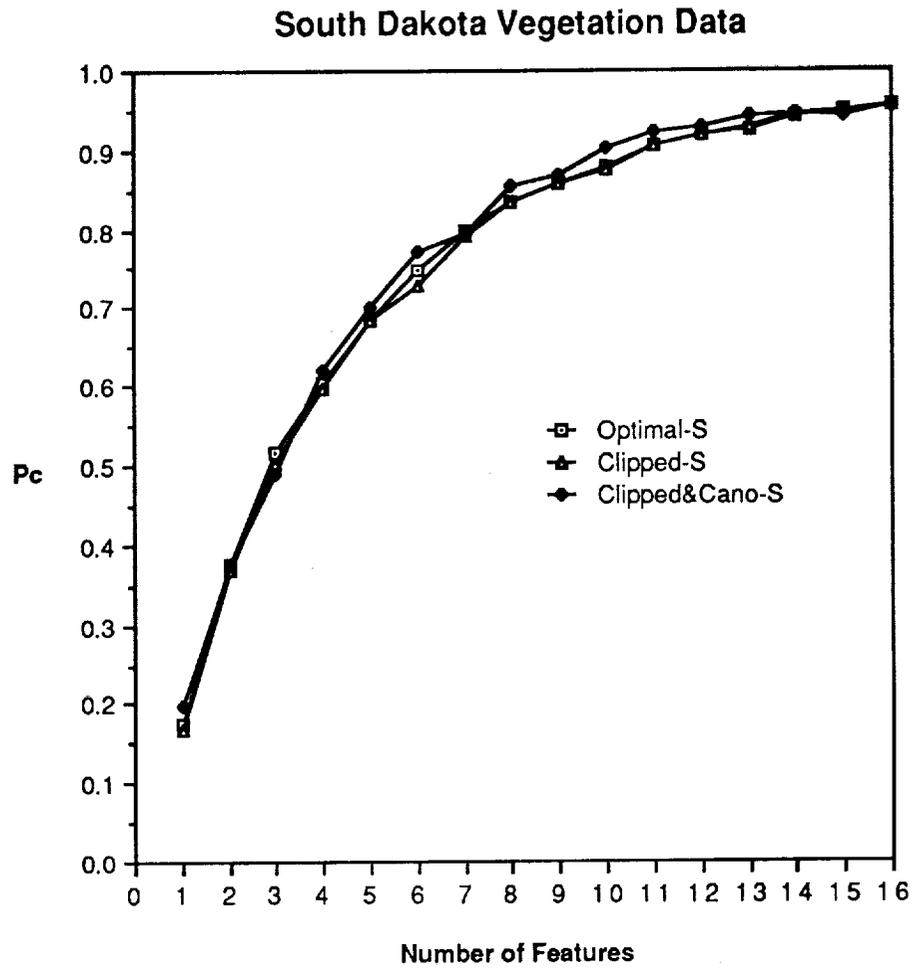
Figure 4.11 Classification Performance for S. Dakota Multitemporal Data Set

4.2 Soil Data

In addition to the above FSS vegetation data, a soil data base with 571 soil samples collected by Eric Stoner [45] in 1979 was acquired to test the system. The soil reflectance functions were measured by an EXOTECH-C spectrometer in the laboratory. In this research, five data sets grouped by soil order, organic matter content #1, organic matter content #2, Iron oxide content and soil texture [46-50] were formed respectively to test the spectral feature design system. They are designated as data sets SO, OM1, OM2, IO and ST respectively. It should be noted that the same soil samples are used in the data sets, but they are only grouped differently into classes. The soil data set designated as organic matter content #1 is from the soil orders Mollisol and Alfisol [48] only, while the soil data set designated as organic matter content #2 is from all soil orders. These 5 soil data sets are shown in Table 4.2

Table 4.2(a) shows the 10 soil orders in American Soil Taxonomy [48]. Since the total numbers of sample functions for Spodosol, Vertisol, Histosol and Oxisol are very limited, in this research, these soils are not used to form the data set SO. Only the data in the first 6 soil orders are included in SO. Table 4.2(b), (c) and (d) indicate the ranges of organic matter content #1, organic matter content #2 and iron oxide content respectively. Six classes are chosen in these 3 data sets: OM1, OM2 and IO. Table 4.2(e) shows the 6 soil texture classes used in data set ST where some of the classes consist of more than one soil texture group. For example, class 1 in data set ST includes clay and silty clay; class 2 includes sandy clay loam, clay loam and silty clay loam; etc.

The results of these 5 soil data sets are shown in Figure 4.12 to 4.16. Taking a general view of these graphs, it is found that the cumulative

performances of these soil data sets are less like a standard error function compared to those found in vegetation data sets (referring to Figure 4.1 to 4.11). The reason for this is that the total numbers of sample functions used to estimate the covariance matrices in the soil data sets are very limited, from a little more than the dimensionality in data set OM1, that is, 255 sample functions with dimensionality 200, to about 2.5 times the dimensionality in SO, OM2, IO and ST, that is about 500 sample functions for each data set; while on the other hand at least 8 times the dimensionality are available in the vegetation data sets. For example, the smallest data set K1 has 832 sample functions with dimensionality 100 and data sets other than K1 have more than 1000 sample functions to estimate the covariance matrix. Therefore, the estimates of the covariance matrices for the vegetation data sets are likely to be much more accurate than those for the soil data sets. The subsequent Gaussian model thus becomes more valid for the vegetation data and the cumulative classification curves are more like a standard error function.

Furthermore, Figure 4.12 to 4.16 show that the infinite clipped optimal functions are very effective to extract the information for soil classification. For instance, Figure 4.12 to 4.13 indicate that using the first 16 infinite clipped optimal functions, over 90% accuracy can be achieved while Figure 4.14 to 4.16 tell that over 85% accuracy is obtained. Due to the limited sample size for each of the soil data sets, different degrees of the Hughes phenomenon occur. Figure 4.12 to 4.14 show that canonical analysis improves the performance for the first 5 features while Figure 4.15 to 4.16 show that improvement is possible up to the first 7 features.

Table 4.2 Soil Data Sets :

(a) SO by Soil Order
Sample size for the first 6 classes : 479

class #	Order Name	# of Sample Functions
1	Mollisol	154
2	Alfisol	113
3	Entisol	78
4	Aridisol	52
5	Ultisol	45
6	Inceptisol	37
7	Spodosol	30
8	Vertisol	11
9	Histosol	8
10	Oxisol	11
11	Unclassified	32

(b) OM1 by Organic #1
Soil from Mollisol and Alfisol only. Sample size : 255

Class #	Organic Matter Range %	# of Sample Functions
1	0.11 ~ 1.5	51
2	1.5 ~ 2.0	54
3	2.0 ~ 2.5	33
4	2.5 ~ 3.5	45
5	3.5 ~ 5.0	39
6	5.0 ~ 10.12	33

(c) OM2 by Organic #2
Soil from all orders. Sample size : 514

Class #	Organic Matter Range %	# of Sample Functions
1	0.08 ~ 1.0	82
2	1.0 ~ 2.0	135
3	2.0 ~ 3.0	120
4	3.0 ~ 4.0	54
5	4.0 ~ 6.0	59
6	6.0 ~ 84.79	64

Table 4.2, continued

(d) IO by Iron Oxide Content
Sample size : 467

Class #	Iron Oxide Range %	# of Sample Functions
1	0.02 ~ 0.4	102
2	0.4 ~ 0.6	73
3	0.6 ~ 0.8	69
4	0.8 ~ 1.2	105
5	1.2 ~ 1.6	52
6	1.6 ~ 25.6	66

(e) ST by Soil Texture
Total sample size : 483 excluding the unclassified

Class #	Soil Texture Group/Groups	# of Sample Function	Class Sample Size
1	Clay	19	
	Silty Clay	21	40
2	Sandy Clay Loam	6	
	Clay Loam	25	
	Silty Clay Loam	32	63
3	Coarse Sand	3	
	Large Coarse Sand	6	
	Sand	13	
	Large Sand	16	
	Large Fine Sand	18	
	Fine Sand	20	76
4	Coarse Sandy Loam	5	
	Very Fine Sandy Loam	12	
	Sandy Loam	24	
	Fine Sandy Loam	52	93
5	Loam	68	68
6	Silt	4	
	Silt Loam	139	143
7	Unclassified	88	88

Soil Data Set Grouped by Soil Order
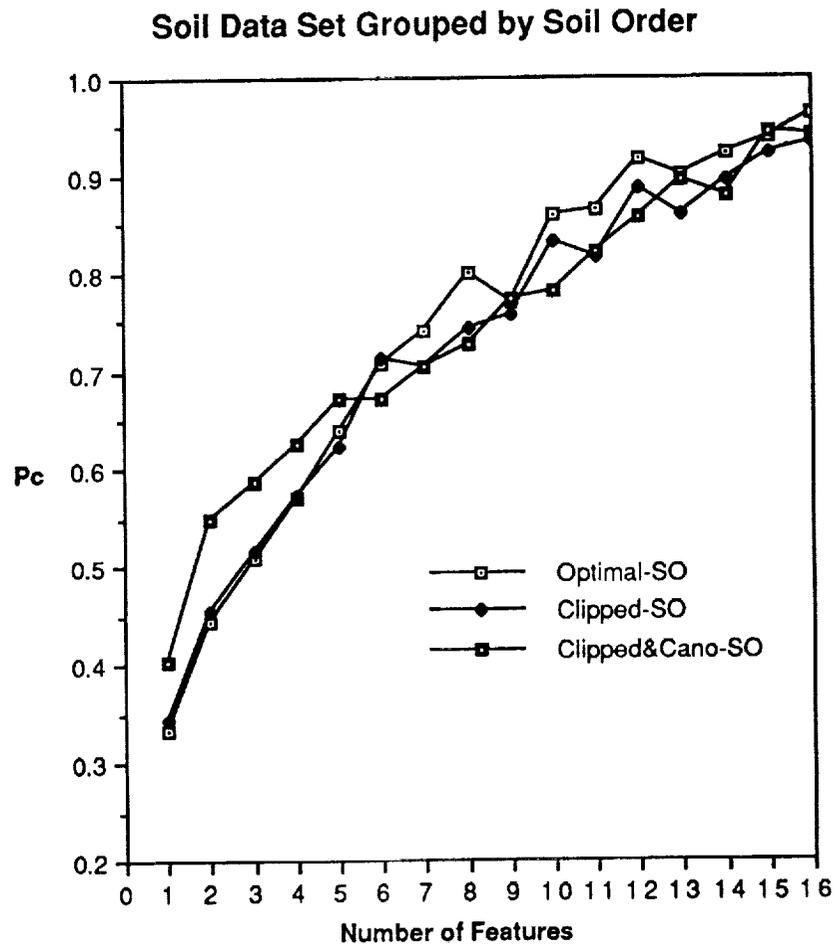
Figure 4.12 Classification Performance for Soil
Data Grouped by Soil Order

Soil Data Set Grouped by Organic Matter#1
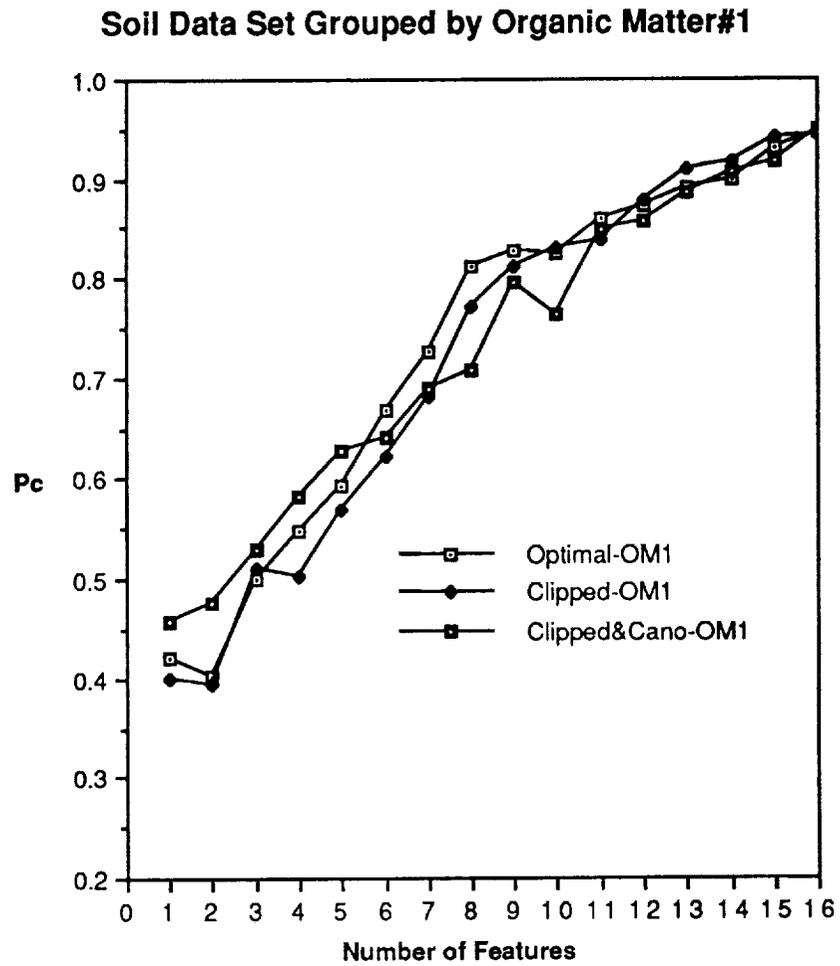
Figure 4.13   Classification Performance for Soil Data Grouped by Organic #1

Soil Data Set Grouped by Organic Matter#2
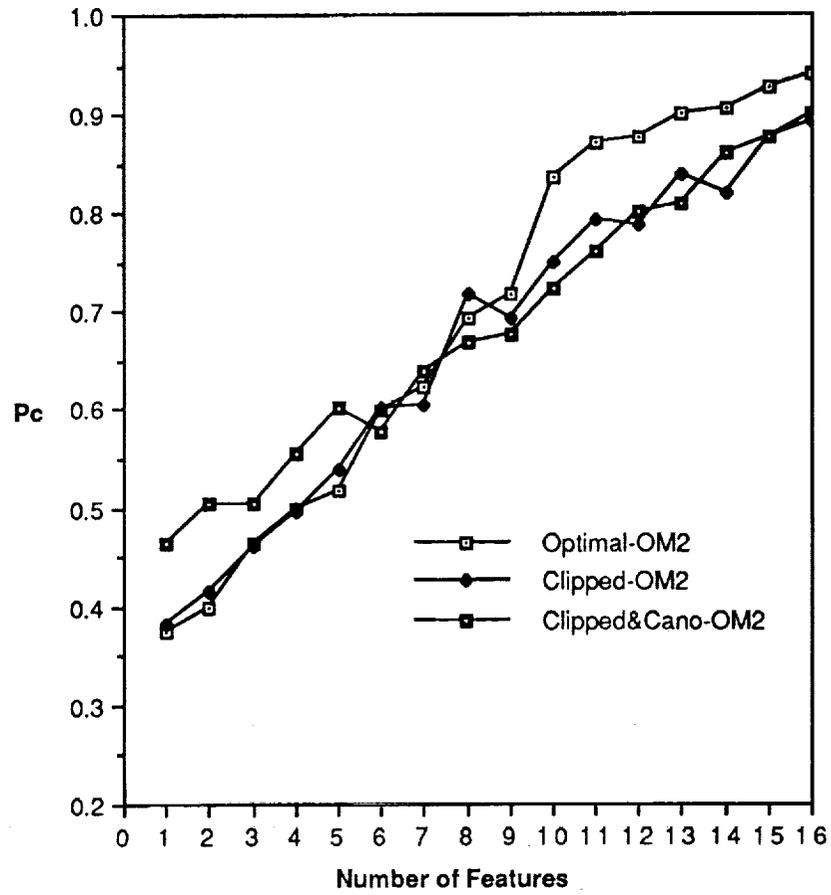
Figure 4.14   Classification Performance for Soil
Data Grouped by Organic #2

Soil Data Set Grouped by Iron Oxide Content
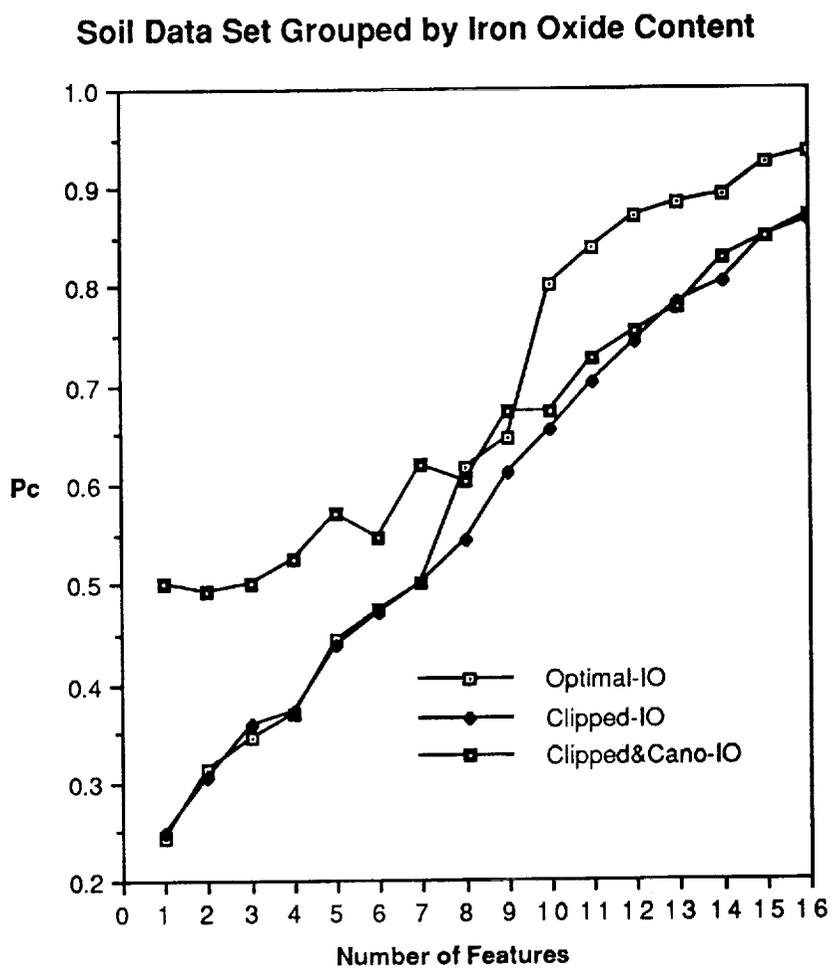
Figure 4.15   Classification Performance for Soil Data Grouped by Iron Oxide

Soil Data Set Grouped by Soil Texture
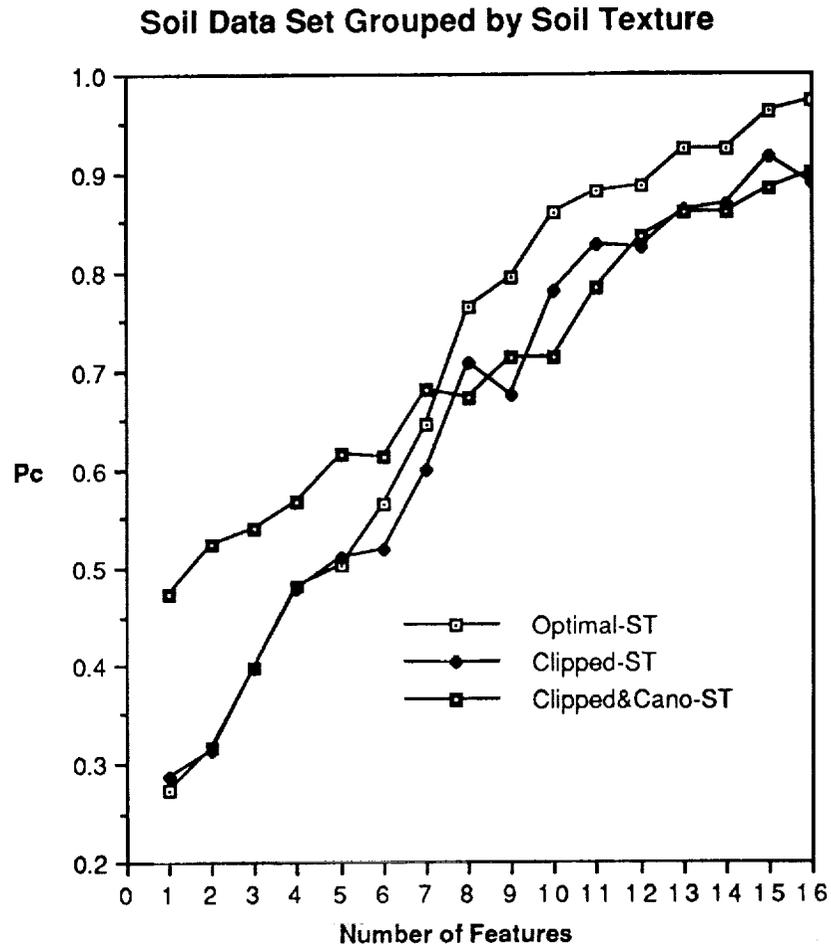
Figure 4.16   Classification Performance for Soil Data Grouped by Soil Texture

4.3 Hughes Phenomenon

In 1968, Hughes [42] showed theoretically that the mean recognition accuracy for the statistical pattern classifiers did not always increase as the measurement complexity increased so long as the number of training samples was fixed and finite. This result was experimentally demonstrated in a remote sensing context by Fu, Landgrebe and Phillips [43] in 1969. The conclusion of these investigations was that for a fixed number of training samples, there is an optimal measurement complexity. More complexity is undesirable from the standpoint of expected classification accuracy.

Kalayeh, Muasher and Landgrebe [51,52] developed a criterion to predict the occurrence of the Hughes phenomenon. They suggested that a number of sample functions equal to about 8 to 10 times the dimensionality must be available for the ensemble in order to avoid the Hughes phenomenon.

In this section, four experiments are described to show that the Hughes phenomenon did occur in the data sets with limited training samples. The data sets K1 and N2 were chosen for this purpose because K1 has the least training samples (referring to Table 1.1) among all vegetation data sets and N2 (referring to Figure 4.5) indicated some possibility for the occurrence of the Hughes phenomenon. Tables 4.3(a) to (d) show the data used for these 4 experiments and Figures 4.17 to 4.20 show the results. In the above tables and figures, K1H and N2H are the data sets with about one half of the original training samples while K1Q and N2Q represent those with approximately one quarter of the training samples.

Figure 4.17 and 4.18 show that for data set K1, the Hughes phenomenon has occurred (referring to Figure 4.1). If the size of the training samples is reduced to half or even to quarter, the effect of this phenomenon becomes more and more serious. On the other hand, for data set N2, there is no Hughes phenomenon (referring to Figure 4.5). If the size of the training samples becomes one half of the original N2, the Hughes phenomenon might or might not occur. Figure 4.19 indicates that for data set N2, reducing the size of the training samples to approximately one half, that is 630 samples with dimensionality 100, the estimate of covariance matrix is still accurate enough, and the Hughes phenomenon does not occur.

However, if the training size of the data set N2 is reduced to one quarter, the Hughes phenomenon does occur. Figure 4.20 says that the optimal number of features in this data set N2Q with 315 training samples is only 2. The maximal classification accuracy that can be achieved is about 85%. Furthermore, more than 2 features used for classification would not help the performance and in some cases even reduce the accuracy.

The four experiments in this section indicate that for data set K1, more than 832 samples are needed in order to avoid the effect of Hughes phenomenon; on the other hand, for data set N2, 1239 samples are enough to accurately estimate the covariance matrix. From the classification performances of data sets K1, K2, K3 and N1, shown in Figure 4.1 to 4.4, it is suggested that more than 15 times dimensionality sample functions may be required to avoid the effect of the Hughes phenomenon.

Table 4.3   Data Sets Used to Test the Occurrence of the Hughes Phenomenon :

(a) Kansas September Data With Half Training Samples : Data Set K1H

K1H	Winter Wheat	Summer Fallow	Grain Sorghum	Total Samples
Training	70	200	140	410
Testing	71	214	137	412
Total	141	414	277	832

(b) Kansas September Data With Quarter Training Samples : Data Set K1Q

K1Q	Winter Wheat	Summer Fallow	Grain Sorghum	Total Samples
Training	35	100	70	205
Testing	106	314	207	627
Total	141	414	277	832

(c) North Dakota June Data With Half Training Samples : Data Set N2H

N2H	Spring Wheat	Summer Fallow	Natural Pasture	Total Samples
Training	400	150	80	630
Testing	387	141	81	609
Total	787	291	161	1239

(d) North Dakota June Data With Quarter Training Samples : Data Set N2Q

N2Q	Spring Wheat	Summer Fallow	Natural Pasture	Total Samples
Training	200	75	40	315
Testing	587	216	121	924
Total	787	291	161	1239
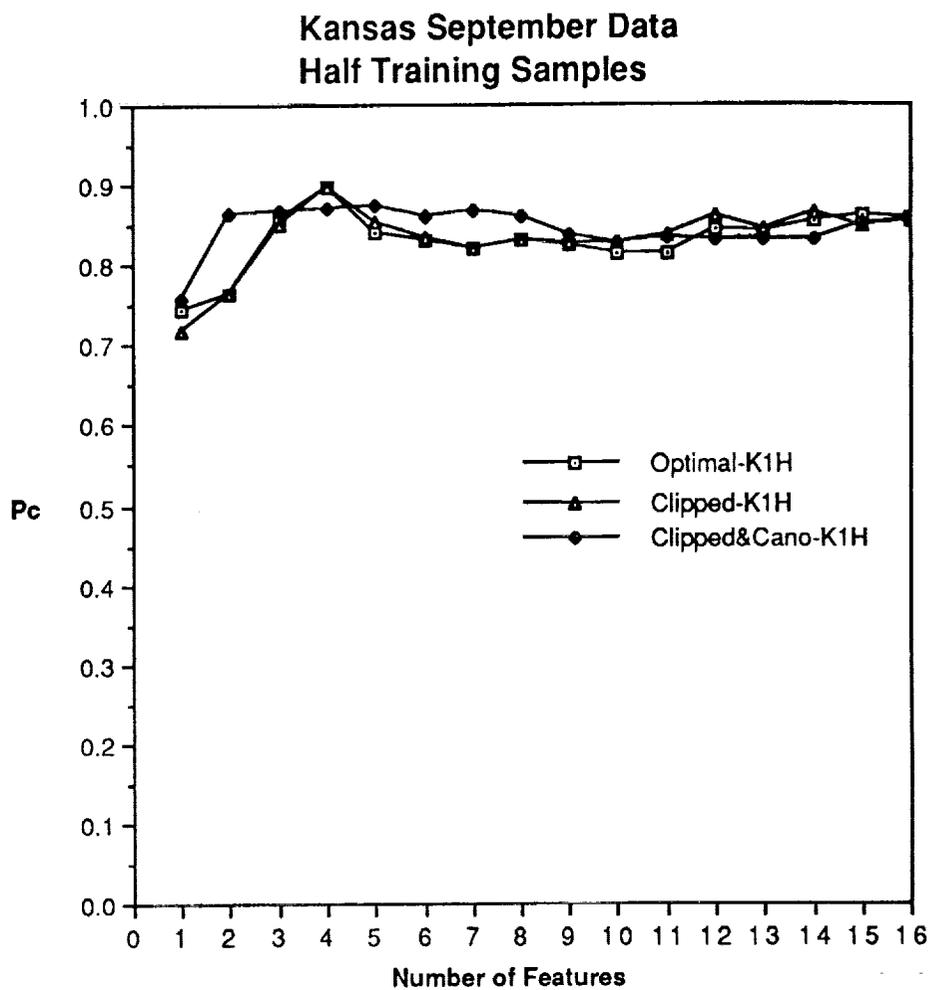
**Figure 4.17 First Experiment of the Hughes Phenomenon :
Data Set K1H**
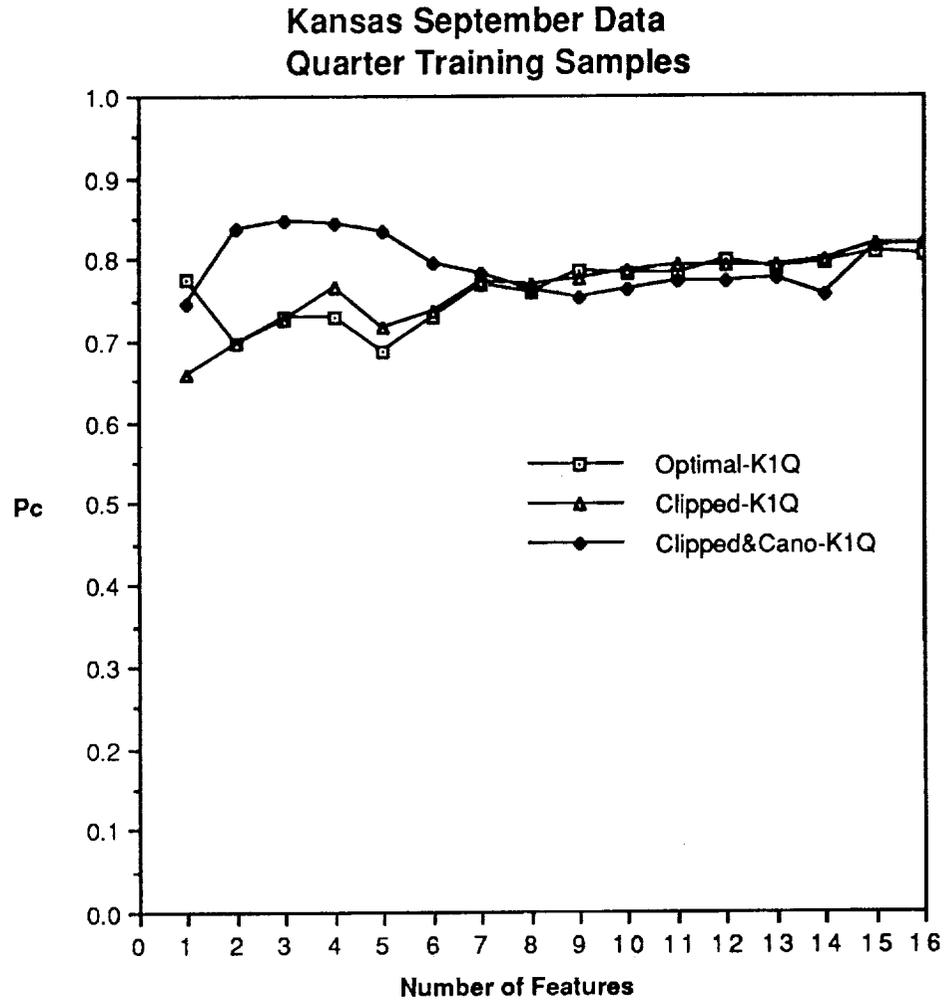
**Figure 4.18   Second Experiment of the Hughes Phenomenon :
Data Set K1Q**

North Dakota June Data
Half Training Samples
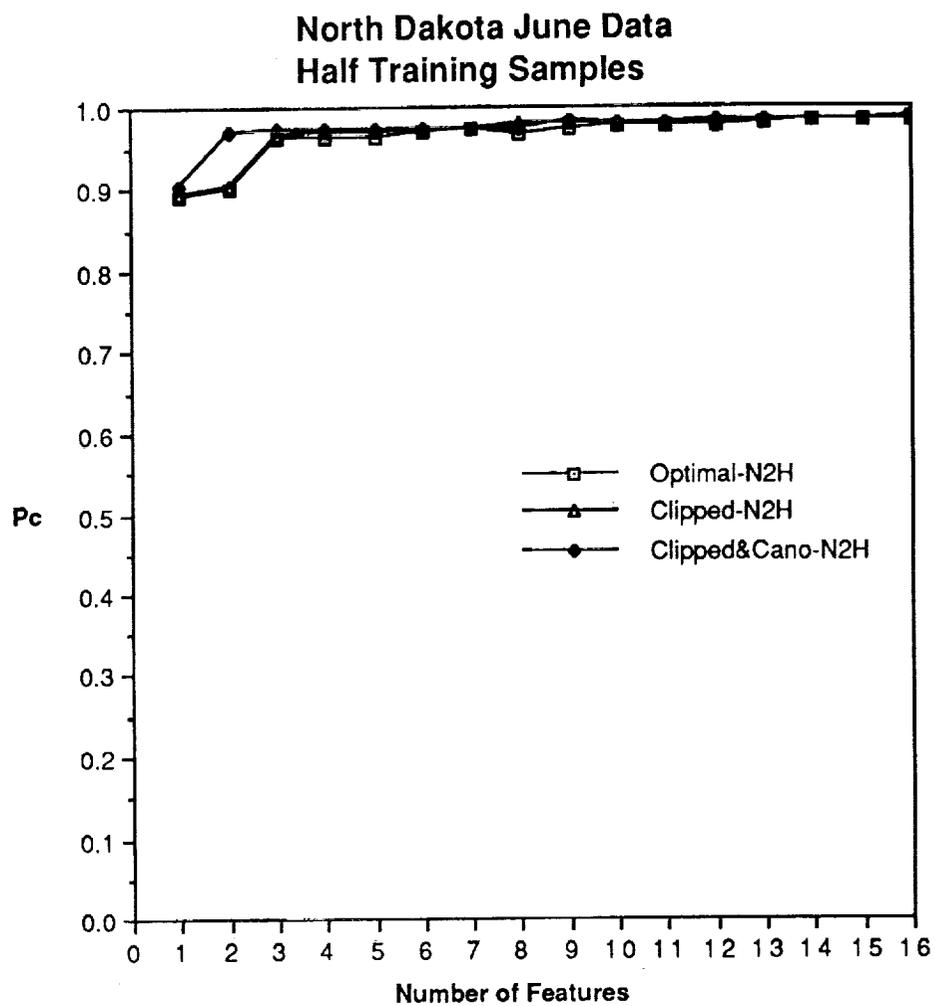
Figure 4.19 Third Experiment of the Hughes Phenomenon :
Data Set N2H

**North Dakota June Data
Quarter Training Samples**
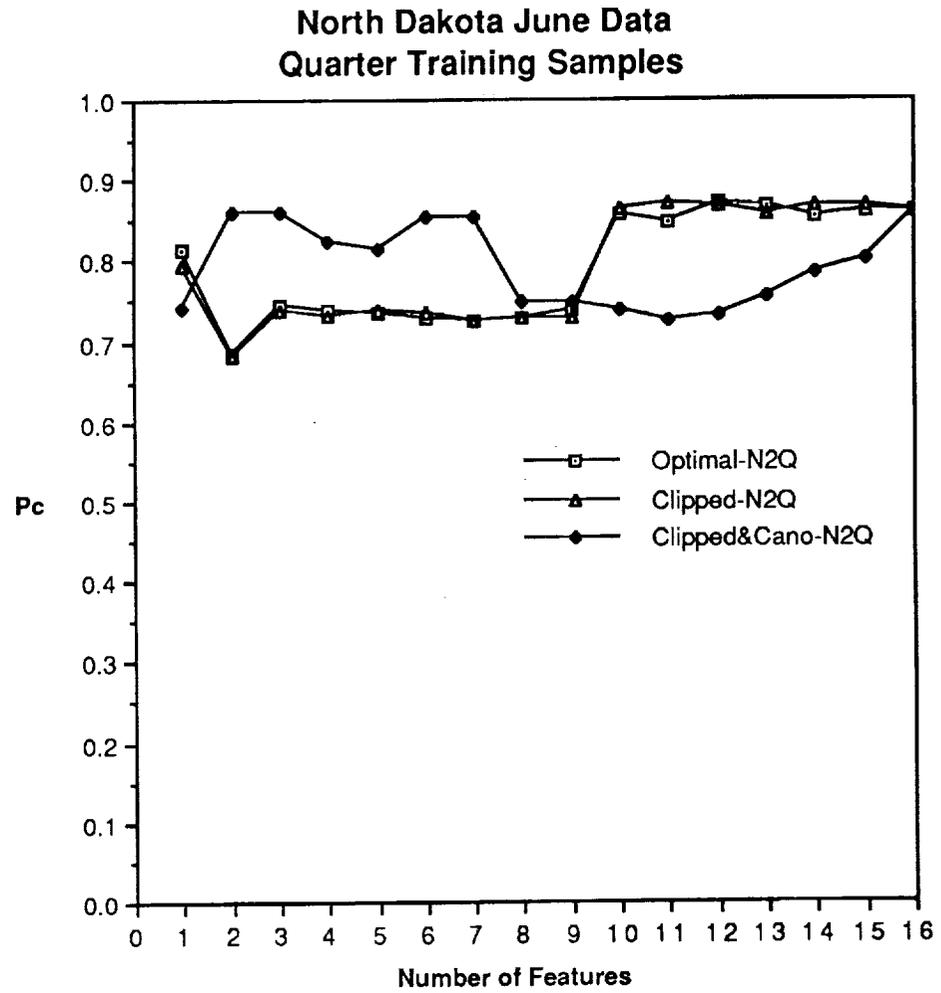
Figure 4.20   Fourth Experiment of the Hughes Phenomenon :
Data Set N2Q

4.4 Signal to Noise Ratio Considerations

In the previous sections, the classification results obtained by using the spectral features developed in this research are presented for 100 dimensional FSS vegetation data and 200 dimensional Exotech-C soil data. It is found (referring to Figure 4.1 to 4.16) that about 10 to 1 compression ratio can be achieved while maintaining satisfactory classification accuracy. One question an Earth scientist user of the algorithm may have is that the 10 to 1 downlink data rate reduction is not at a severe cost to the usefulness of the data. Thus, in this section, we will discuss the data volume reduction issue from the Earth scientist point of view, that is, from signal-to-noise ratio considerations.

Weighted Karhunen-Loeve transform rotates the original N-dimensional signal space to a more favorable orientation. This orientation is one in which the source energy is redistributed such that a larger percentage of the energy is distributed over fewer coordinates. Table 4.4 and Figure 4.21 show how the source energy is redistributed over the first 25 transformed coordinates for 100 dimensional vegetation data set K2.

In Table 4.4, the first row shows that the magnitude of the total source energy is 3497, which is the sum of all eigenvalues; Further, the mean square representation error (MSE) and percent mean square representation error (%MSE) are 3497 and 100% respectively if 'none' of the optimal feature is used to transform the data. The second row indicates that the magnitude of the first eigenvalue is 2779.8; If the first optimal feature is used to transform the data, the representation error and percent representation error will be 717 and 20.5% respectively, that is, the first transformed coordinate contains about 79.5%

source energy in it. Similarly, it can be found that using the first 2 optimal features, about 97.5% of the total source energy can be preserved, and using the first 10 optimal features to transform the data in the measurement space, the percent mean square representation error, that is 0.17%, is indeed negligible. Figure 4.21 shows graphically how fast the representation error can be reduced by using the first few optimal features. It should be noticed that the representation error (MSE) is plotted in logarithmic scale.

The practical values of the signal to noise ratio in a typical remote sensing system are from 50 to 200 in most of the 0.4 to 2.5 μm spectrum range [1]. This indicates that the maximal noise level in the system is only 1/50, that is, 2%. Since using the first 10 optimal features derived from the Weighted K-L transform preserves almost all the signal energy in the original measurement space; Further, the representation error level is 0.17% which is much lower than the noise level in the system. Hence, the effect on the signal to noise ratio due to compression is quite limited even as the signal to noise ratio is down to 20. Therefore, a data volume reduction by a factor of 10 is achieved with essentially no loss of information.

Table 4.4 Mean Square Representation Error for Data Set K2

Eigenvalue	Magnitude of Eigenvalue	Mean Square Error	% Mean Square Error
0	3497.0691	3497.0691	100.0000
1	2779.8821	717.1870	20.5082
2	627.0327	90.1543	2.5780
3	39.0218	51.1325	1.4622
4	18.4108	32.7217	0.9357
5	14.0425	18.6792	0.5341
6	4.9193	13.7599	0.3935
7	2.5450	11.2149	0.3207
8	1.8422	9.3727	0.2680
9	1.7561	7.6166	0.2178
10	1.3731	6.2435	0.1785
11	0.8927	5.3508	0.1530
12	0.8225	4.5283	0.1295
13	0.6291	3.8993	0.1115
14	0.4818	3.4175	0.0977
15	0.4498	2.9676	0.0849
16	0.3778	2.5898	0.0741
17	0.3469	2.2429	0.0641
18	0.3266	1.9163	0.0548
19	0.2328	1.6835	0.0481
20	0.2192	1.4643	0.0419
21	0.1696	1.2947	0.0370
22	0.1499	1.1448	0.0327
23	0.1268	1.0181	0.0291
24	0.1174	0.9006	0.0258
25	0.0904	0.8103	0.0232

Mean Square Representation Error
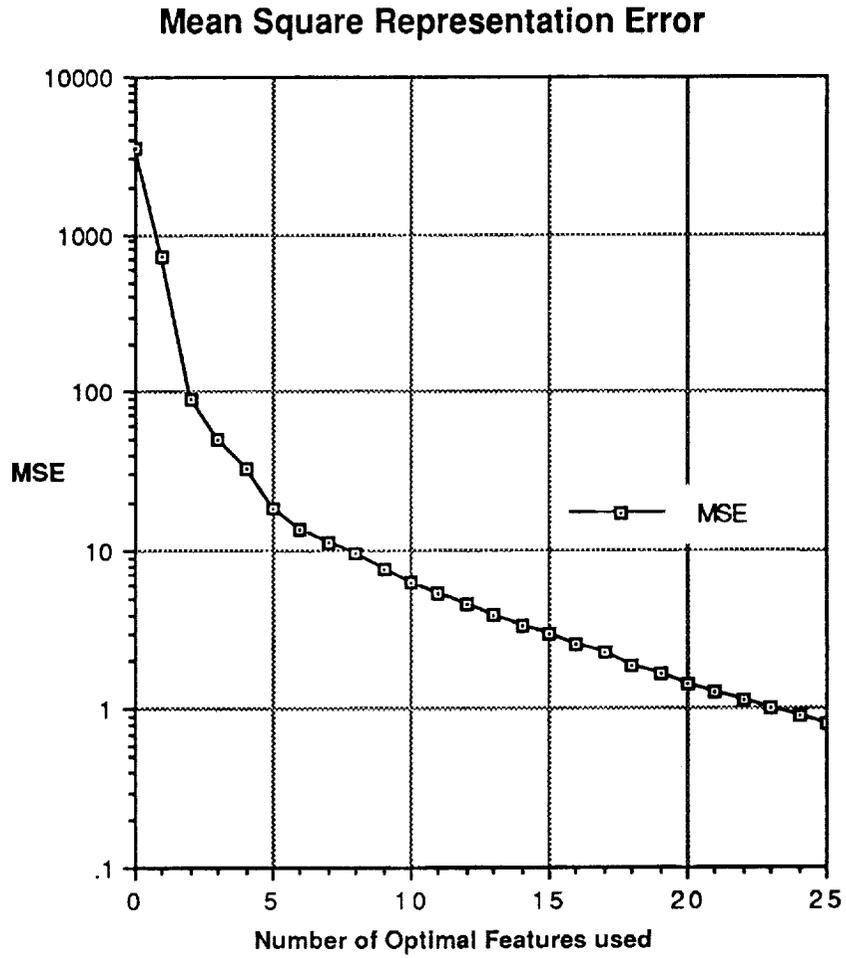
Figure 4.21   Mean Square Representation Error for Data Set K2

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

The fundamental objective of this research is to develop an objective and practical spectral feature design technique for high dimensional multispectral data. In this thesis, four spectral feature design techniques have been developed. Two of them, non-overlapping band feature selection algorithm and overlapping band feature selection algorithm, are derived from the spectral dominancy concept of the optimal functions; the other two, Walsh function approach and infinite clipped optimal function approach, are derived from the spectral similarity concept of the optimal functions. These four approaches have been proved effective for data compression and classification purposes in high dimensional multispectral data.

A comparison among these four techniques indicates that the infinite clipped optimal function approach is the best scheme since the features are easiest to find and their classification performance is the best under the same compression requirement. This technique approximates the spectral structure of the optimal features via infinite clipping and results in transform coefficients which are either +1, -1 or 0. Therefore the necessary processing can be easily implemented on-board the spacecraft by using a set of programmable adders that operate on the grouping instructions received from the ground station.

After the preprocessed data has been received, canonical analysis is further used to find the best set of features under the criterion that maximal class separability is achieved

Both vegetation and soil data have been tested in this research. For vegetation data, four sets of multitemporal multispectral vegetation data collected in Kansas, North Dakota, Iowa and South Dakota respectively with 9 to 42 information classes in 1976 to 1979 are used to test the spectral feature design system. One spatially and temporally combined data set is also formed by combining the Kansas and North Dakota Data sets to test the robustness property of the scheme. The results indicate that the system is not overly sensitive to spatial and temporal variation.

Furthermore, a soil data base collected by Eric Stoner in 1979 was also acquired and used to test the system. In this research, five different soil data sets grouped by the soil order, organic content #1, organic content #2, iron oxide content and soil texture are formed. The classification performances are then found. It is shown that soil order, organic content percentage, iron oxide content percentage and soil texture can be delineated and predicted by the proposed technique.

It is concluded that the infinite clipped versions of the first 16 optimal functions derived from the Weighted Karhunen-Loeve Transform have excellent classification performance. Further signal processing by canonical analysis increases the compression ratio while retains the classification accuracy. The overall probability of correct classification of the proposed system is over 90% while providing for a reduced downlink data rate by a factor of 10.

5.2 Recommendations

The spectral feature design system developed in this research has been demonstrated for the FSS vegetation data and the Exotech-C soil data. In the future, it is proposed to test AVIRIS and HIRIS data. The following procedure is recommended :

(A)   Pre-Flight Stage :

(1)   Collect enough representable samples from all reference sources available, for example, the field data base collected in the past, to form the ensemble of a specific problem (Ground Truth Gathering)

(2)   Calculate the mean vector and covariance matrix of this ensemble

(3)   Find the eigenvectors of the covariance matrix

(4)   Run the spectral feature design system on the ground to find the grouping coefficients (either +1, -1, or 0)

(B)   On-Board Preprocessing Stage :

(5)   Send up these grouping coefficients (instructions) to the spacecraft for on-board data preprocessing

(C)   Post-Flight Stage :

(6)   Receive the preprocessed low dimensional data

(7)   Run the spectral feature design system on the ground to find the canonical features

(8)   Use these canonical features to further transform the received data into the final signal space where the data classification is performed

In this procedure, there are basically 3 processing stages involved : pre-flight stage, on-board preprocessing stage and post-flight stage. The pre-flight stage, which consists of step 1 to step 4, is used to gather ground truth information, estimate ensemble statistics and find appropriate grouping coefficients from one of the four developed schemes. This stage would be done before the data take by the aids of aerial photography, topographical maps, historical information, field data base collected in the past or other reference sources available. One more comment about this stage is the problem of the sample size, it is suggested from the experience in this research that the total number of samples used to estimate the ensemble statistics needs to be at least 15 times their signal dimensionality in order to accurately estimate the covariance matrix.

The second stage, on-board preprocessing stage, which contains step 5, performs band groupings on board the spacecraft, either summing (+1), subtracting (-1) or omitting (0) bands for each spectral function according to the grouping instructions sent by the ground user. Since this data preprocessing stage would be done on board the spacecraft, from implementation point of view, the algorithm simplicity is then required and important. The spectral feature design system developed in this research makes this simplicity possible. Figure 1.1 shows how the data preprocessing can be implemented on board the spacecraft by a set of programmable adders.

Finally, the post-flight stage, which includes step 6 to step 8, is applied to further process the received transformed data such that the maximal class

separability is achieved. Since this stage and the pre-flight stage would be done at the ground station, the algorithm simplicity is therefore less important than that in the on-board preprocessing stage. Hence, it might be more effective to use the overlapping band feature selection algorithm to design the features in some future situations although it's the most complex among the four techniques developed in this research.

LIST OF REFERENCES

[1] A. F. H. Goetz and M. Herring, "The High Resolution Imaging Spectrometer (HIRIS) for EOS," Proceedings of IGARSS '87 Symposium, pp 367-372, Ann Arbor, 18-21 May 1987.

[2] C. L. Korb, "Physical Considerations in the Channel Selection and Design Specification of an Airborne Multispectral Scanner," Purdue Centennial Year Symposium on Information Processing, Purdue University, West Lafayette, Indiana, pp 646-657, 1969

[3] R. L. Hulstrom, "Spectral Measurements and Analyses of Atmospheric Effects on Remote Sensor Data," Proceedings of the Society of Photo-Optical Instrument Engineers, Vol. 51, Scanners and Imagery Systems for Earth Observation, San Diego, California, pp 90-100, 1974

[4] J. L. Harnage and D. A. Landgrebe, "Landsat-D Thematic Mapper Technical Working Group - Final Report," NASA/Johnson Space Center Report No. JSC-90797, Houston, Texas, 1975

[5] G. A. May and G. W. Petersen, "Spectral Signature Selection for Mapping Unvegetated Soils," Remote Sensing of Environment, Vol. 4, pp 211-220, 1975.

[6] L. A. Bartolucci, B. F. Robinson, and L. F. Silva, " Field Measurements of the Spectral Response of Natural waters, " Photogrammetric Engineering and Remote Sensing, Vol. 43, pp 595-598, 1977.

[7] D. A. Landgrebe, L. L. Biehl and W. R. Simmons, "An Empirical Study of Scanner System Parameters, " IEEE Transaction on Geoscience Electronics, Vol. GE-15, pp 120-130, 1977

[8] M. E. Coggeshall and R. M. Hoffer, "Basic Forest Cover Mapping Using Digitized Remote Sensor Data and Automatic Data Processing Techniques," LARS Information Note 030573, Laboratory for Application of Remote Sensing, Purdue University, West Lafayette, Indiana, 1973

[9] R. K. Vincent and F. Thompson, "Spectral Compositional Imaging of Silicate Rocks, " J. of Geophysics, Res., Vol. 77, pp 2465-2472, 1972.

[10] W. A. Malila, J. M. Gleason, and R. C. Cicone, "Multispectral System Analysis through Modeling and Simulation," Proceedings of 11th International Symposium on Remote Sensing, Ann arbor, Michigan, pp 1319-1328, 1977.

[11] P. H. Swain and S. M. Davis, *Remote Sensing : the Quantitative Approach,* McGraw-Hill, New York, 1978.

[12] E. R. Wiswell, " Analytical Techniques for the Study of Some Parameters of Multispectral Scanner Systems for Remote Sensing," Ph.D thesis, Purdue University, West Lafayette, Indiana, 1978.

[13] R. Ash, *Information Theory*, Wiley Interscience, New York, 1967

[14] D. J. Wiersma and D. A. Landgrebe, "Analytical Design of Multispectral Sensors," IEEE Transactions on Geoscience and Remote Sensing, Vol. GE-18, No. 2, pp 180-189,1980.

[15] D. J. Wiersma, "The Analytical Design of Spectral Measurements for Multispectral Remote Sensor System," Ph.D. dissertation, Purdue University, 1979

[16] K. Karhunen, "Uber Lineare Methoden in der Wahrscheinlichkeitsrechnung," American Acad. Sci. Fennicade, Ser. A, I, Vol. 37, pp 3-79, (Translated by RAND Corp., Santa Monica, Calif., Rept. T-131, August 1960).

[17] M. Loeve, " Probability Theory," Princeton, N. J., Van Nostrand, 1963.

[18] H. L. VanTree, *Detection, Estimation, and Modulation Theory-Part I,* New York: Wiley, 1968

[19] L. E. Franks, *Signal Theory*, Dowden & Culver, 1981.

[20] K. G. Beauchamp, *Walsh Functions and their Applications,* Academic Press, New York, 1975

[21] B. Pyron and F. Williams, Jr., "Signal Processing by Infinite Clipping and Related Techniques," Georgia Institute of technology, Atlanta, Georgia, Final Report, Project A-727, U.S. Government Contract DA 49-092-ARO-21, April 1964

[22] B. Noble and J. W. Daniel, *Applied Linear Algebra,* Prentice-Hall, N. J., 1977.

[23] L. L. Biehl, "Listing of LARSPEC Formatted Field Research Spectroradiometer-Multiband Radiometer, " Internal Paper, Laboratory of Application of Remote Sensing, Purdue University, West Lafayette, Indiana, Jan. 1983.

[24] B. H. Bharucha and T. T. Kadota, "On the representation of Continuous Parameter Process by a Sequence of Random Variables," IEEE Transaction on Information Theory, Vol. IT-16, pp 139-141, 1970.

[25] S. Watanabe, "Karhunen-Loeve Expansion and Factor Analysis, Theoretical Remarks and Applications," Proceedings of the Four Prague Conference on Information Theory, Statistical Decision Functions, and Random Processes," Prague, pp 635-660, 1965.

[26] Y. T. Chien and K. S. Fu, "On the Generalized Karhunen-Loeve Expansion," IEEE Transaction on Information Theory, Vol. IT-13, pp 518-520, July 1967.

[27] J. T. Oden, *Applied Functional Analysis*, Prentice-Hall, N. J., 1979.

[28] R. W. A. Scarr, "Zero Crossing as a Means of Obtaining Spectral Information in Speech Analysis," IEEE Transaction on Audio and Electroacoustics, Vol. AU-16, No. 2, pp 247-255, June 1968.

[29] G. D. Ewing and J. F. Taylor, " Computer Recognition of Speech Using Zero-Crossing Information, " IEEE Transactions on Audio Electroacoustics, Vol. AU-17, pp37-40, Mar. 1969

[30] R. J. Niederjohn, M. W. Krutz and B. M. Brown, " An Experimental Investigation on the Perceptual Effects of Altering the Zero-Crossing of a Speech Signal ", IEEE Transaction on Acoustics, Speech, and Signal Processing, Vol. ASSP-35, No. 5, pp618-625, May 1987

[31] F. E. Bond and C. R. Cahn, "On Sampling the Zeros of Bandwidth Limited Signals," IRE Transaction on Information Theory, Vol. IT-4, pp 110-113, September 1958.

[32] M. S. Scivier and M. A. Fiddy, "Phase Ambiguities and the Zeros of Multidimensional Band-limited Functions," Optical Society of America, Vol. 2, No. 5, pp 693-697, May 1985.

[33] R. J. Niederjohn, " A Mathematical Formulation and Comparison of Zero-Crossing Analysis Techniques Which Have Been Applied to Automatic Speech Recognition," IEEE Transaction on Acoustics, Speech and Signal Processing, Vol. ASSP-23, No. 4, pp 373-380, August 1975.

[34] K. Fukunaga, *Introduction to Statistical Pattern Recognition,* New York: Academic, 1972

[35] C-C Thomas Chen and D. A. Landgrebe, " Spectral Feature Design for Data Compression in High Dimensional Multispectral Data ", 1987 International Geoscience and Remote Sensing Symposium ('87 IGARSS), pp 685-690, May 1987

[36] T.W. Anderson, *An Introduction to Multivariate Statistical Analysis,* New York, Wiley 1978

[37] John A. Richards, *Remote Sensing Digital Image Analysis, An Introduction,* Springer Verlag, pp 206 - 224.

[38] H.M. Lachowski, " Canonical Analysis Applied to Interpretation of Multispectral Scanner Data ", MS Thesis, Pennsylvania State University, University Park, PA, 1973

[39] B.F. Merembeck and F.Y. Borden, " Principal Component and Canonical Analysis for Reduction of Dimensionality of the Large Data Sets", ORSER Technical Report 5-78, Office for Remote Sensing of Earth Resources, Pennsylvania State University, University Park, PA, 1978

[40] B.F. Merembeck, F.Y. Borden, M.H. Podwysocki and D.N. Applegate, " Application of Canonical Analysis to Multispectral Scanner Data ", Proceedings, the 14th Annual Symposium on Applications of Computer Methods in the Mineral Industries, Oct. 4-8, University Park, PA, pp867-879, 1976

[41] B.F. Merembeck & B.J. Turner, "Directed Canonical Analysis and the Performance of the Classifiers under its associated Linear Transformation," IEEE Transaction on Geoscience and Remote Sensing, GE-18, 190-196, 1980.

[42] G.F. Hughes, " On the Mean Accuracy of Statistical Pattern Recognizers ", IEEE Transaction on Information Theory, vol. IT-14, no. 1, pp55-63, 1968

[43] K.S. Fu, D.A. Landgrebe and T.L. Phillips, " Information Processing of Remotely Sensed Agriculture Data ", Proceedings of IEEE, vol. 57, no. 4, pp639-653, 1969

[44] Davenport and Root, *Random Signals and Noise,* McGraw-Hill Book Co., 1958

[45] Eric Stoner, " Physicochemical, Site, and Bidirectional Reflectance Factor Characteristics of Uniformly-Moist Soils ", Ph.D. Dissertation, Department of of Agronomy, Purdue University, 1979

[46] Soil Survey Staff, " Soil Taxonomy - A basic system of soil classification for making and interpreting soil survey ", Soil Conservative Service, US Department of Agriculture, Agriculture Handbook No. 436, Washington, DC, 1975

[47] Eric Stoner, " Atlas of Soil Reflectance Properties ", Agriculture Experiment Station, Research Bulletin, Purdue University, West Lafayette, Indiana, 1979

[49] Marion F. Baumgardner, LeRoy F. Silva, Larry L. Biehl, and Eric R. Stoner, " Reflectance Properties of Soils ", Purdue University, West Lafayette, Indiana, Advances in Agronomy, vol. 38, 1985

[50] R. M. Hoffer and C. J. Johannsen, Remote Sensing Ecology, P.L. Johnson, ed., pp1-29, University of Georgia Press, Athens, 1969

[51] A.J. Rainal, "Zero-Crossing Principle for Detecting Narrowband Signals, " IEEE Transaction on Instrumentation and Measurement, Vol. IM-15, pp38-43, March/June 1966

[52] A.J. Rainal, "Another Zero-Crossing Principle for Detecting Narrowband Signals," IEEE Transaction on Instrumentation and Measurement, Vol. IM-16, pp135-138, June 1967

[53] T. Masuda, H. Miyano, and T. Sadoyama, " The Measurement of Muscle Fiber Conduction Velocity Using a Gradient Threshold Zero-Crossing Method, " IEEE Transaction on Biomedical Engineering, Vol. BME-29, pp673-678, Oct. 1982

[54] H. B. Voelcker, " Zero-Crossing Properties of Angle Modulated Signals, " IEEE Transaction on Communication, Vol. COM-20, pp307-315, June 1972

[55] R. G. Wiley, H. Schwarzlander, and D. D. Weiner, " Demodulation Procedure for Very Wide-band FM, " IEEE Transaction on Communication, Vol. COM-25, pp 318-327, March 1977

[56] R. M. Haralick, " Digital Step Edges from Zero-Crossing of Second Directional Derivatives, " IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, pp 58-68, Jan. 1984

[57] Courant and Hilbert, *Mehods of Mathematical Physics,* International Publisher Ltd, London, 1955

Appendix A   IBM 3083 Macro File

```
/* RUN A FORTRAN PROGRAM USING IMSLSP OR IMSLDP SUBROUTINES */
ARG FN FN1 FN2 FN3 FN4 FN5 FN6 FN7 FN8 FN9 FN10 FN11
LINKTO IMSL
GLOBAL TXTLIB IMSLSP IMSLDP PFORTLIB VSF2FORT CMSLIB
GLOBAL LOADLIB VSF2LOAD
FORTVS2 FN
LOAD FN
FILEDEF 11 DISK FN1 DATA C1
FILEDEF 12 DISK FN2 DATA C1
FILEDEF 13 DISK FN3 DATA C1
FILEDEF 14 DISK FN4 DATA C1
FILEDEF 15 DISK FN5 DATA C1
FILEDEF 16 DISK FN6 DATA C1
FILEDEF 17 DISK FN7 DATA C1
FILEDEF 18 DISK FN8 DATA C1
FILEDEF 19 DISK FN9 DATA C1
FILEDEF 20 DISK FN10 DATA C1
FILEDEF 21 DISK FN11 DATA C1
START
```

Appendix B   Spectral Feature Design System - Program Listing

```
      PROGRAM MV                                                      MV 00010
      PARAMETER(NP2=1551,NP1=100,NP3=NP1*(NP1+1)/2,NF2=10,NF3=5)      MV 00020
      REAL X(NP2,NP1),XM(NP1),VCV(NP3)                                MV 00030
      DATA IFLAG1,XM,VCV/0,NP1*0.0,NP3*0.0/                           MV 00040
C                                                                     MV 00050
C     NP1 :  DIMENSIONALITY OF SAMPLE FUNCTIONS                       MV 00060
C     NP2 :  TOTAL NUMBER OF SAMPLE FUNCTIONS                         MV 00070
C     NP3 :  TOTAL NUMBER OF ELEMENTS IN COVARIANCE MATRIX VCV        MV 00080
C     NF2 :  RAW DATA INPUT FILE STORED IN FORMAT 10F8.3              MV 00090
C     NF3 :  XM & VCV OUTPUT DATA FILE STORED IN FORMAT 5E15.7        MV 00100
C                                                                     MV 00110
C     X   :  RAW DATA ( INPUT )                                       MV 00120
C     XM  :  MEAN VECTOR ( OUTPUT )                                   MV 00130
C     VCV :  COVARIANCE MATRIX STORED IN SYMMETRIC MODE ( OUTPUT )    MV 00140
C     IFLAG1 ------   INTERNAL CHECKING PARAMETER                     MV 00150
C                                                                     MV 00160
C     11 = DATA FILE ;   12 = MV FILE                                 MV 00170
C                                                                     MV 00180
      OPEN(11)                                                        MV 00190
      OPEN(12)                                                        MV 00200
      REWIND 11                                                       MV 00210
      REWIND 12                                                       MV 00220
C                                                                     MV 00230
C     READ IN RAW DATA AND PRINT THE PROGRESS FOR EVERY 100 SAMPLES   MV 00240
C                                                                     MV 00250
      DO 20 ISAMP=1,NP2                                               MV 00260
      K=MOD(ISAMP,100)                                                MV 00270
      IF(K.EQ.0)PRINT*,' NP2 = ',NP2,' ; ISAMP = ',ISAMP             MV 00280
      DO 20 I=1,NP1/NF2                                               MV 00290
   20 READ(11,1)(X(ISAMP,J),J=1+(I-1)*NF2,I*NF2)                      MV 00300
      PRINT*,' DATA READ IN FINISHED '                                MV 00310
    1 FORMAT(10F8.3)                                                  MV 00320
C                                                                     MV 00330
C     FIND THE ENSEMBLE MEAN VECTOR                                   MV 00340
C                                                                     MV 00350
      DO 30 J=1,NP1                                                   MV 00360
      DO 30 I=1,NP2                                                   MV 00370
   30 XM(J)=XM(J)+X(I,J)                                              MV 00380
      DO 40 I=1,NP1                                                   MV 00390
   40 XM(I)=XM(I)/FLOAT(NP2)                                          MV 00400
      PRINT*,' MEAN VECTOR FOUND '                                    MV 00410
C                                                                     MV 00420
C     FIND THE ENSEMBLE COVARIANCE MATRIX AND PRINT THE PROGRESS FOR  MV 00430
C     EVERY 10 DIMENSIONS                                             MV 00440
C                                                                     MV 00450
      DO 50 I=1,NP1                                                   MV 00460
      KX=MOD(I,10)                                                    MV 00470
      IF(KX.EQ.0)PRINT*,I                                             MV 00480
      DO 50 J=1,I                                                     MV 00490
      IND=(I-1)*I/2+J                                                 MV 00500
```

```
      DO 50 K=1,NP2                                              MV 00510
   50 VCV(IND)=VCV(IND)+(X(K,I)*X(K,J)-XM(I)*XM(J))              MV 00520
      DO 60 I=1,NP3                                              MV 00530
   60 VCV(I)=VCV(I)/FLOAT(NP2-1)                                 MV 00540
      PRINT*,' COV. MATRIX FOUND '                               MV 00550
C                                                                MV 00560
C     INTERNAL CHECKING FOR ALGORITHM ACCURACY                  MV 00570
C                                                                MV 00580
      DO 80 I=1,NP1                                              MV 00590
      IND=I*(I+1)/2                                              MV 00600
      IF(VCV(IND).LT.0.0)GO TO 70                                MV 00610
      GO TO 80                                                   MV 00620
   70 WRITE(*,2)I,VCV(IND)                                       MV 00630
    2 FORMAT('ACCURACY OF ALGORITHM IS POOR AT I =',I5,          MV 00640
     +' WHERE VCV(I,I) = ',E15.7)                                MV 00650
      VCV(I)=-VCV(I)                                             MV 00660
      IFLAG1=IFLAG1+1                                            MV 00670
   80 CONTINUE                                                   MV 00680
C                                                                MV 00690
C     PRINT THE COMMENTS FOR ACCURACY                           MV 00700
C                                                                MV 00710
      IF(IFLAG1.GT.0)GO TO 90                                    MV 00720
      PRINT*, ' POSITIVE VARIANCES CHECK DONE '                  MV 00730
      WRITE(*,4)                                                 MV 00740
      GO TO 100                                                  MV 00750
   90 WRITE(*,3)IFLAG1                                           MV 00760
    3 FORMAT(' THERE ARE ',I5,' VARIANCES LESS THAN 0.0 ')       MV 00770
    4 FORMAT(' ALL VARIANCES ARE ">= 0.0", ACCURACY IS GOOD')    MV 00780
C                                                                MV 00790
C     SEND THE RESULTS TO OUTPUT DATA FILE                      MV 00800
C                                                                MV 00810
  100 DO 110 I=1,NP1/NF3                                         MV 00820
  110 WRITE(12,5)(XM(J),J=1+(I-1)*NF3,I*NF3)                     MV 00830
    5 FORMAT(5E15.7)                                             MV 00840
      DO 120 I=1,NP3/NF3                                         MV 00850
  120 WRITE(12,5)(VCV(J),J=1+(I-1)*NF3,I*NF3)                    MV 00860
      STOP                                                       MV 00870
      END                                                        MV 00880


      PROGRAM EV                                                 EV 00010
      PARAMETER(NP1=100,NP3=NP1*(NP1+1)/2,NP5=NP3+NP1,           EV 00020
     +NF2=10,NF3=5)                                              EV 00030
      REAL XM(NP1),VCV(NP3),VCVF(NP1,NP1),D(NP1),                EV 00040
     +Z(NP1,NP1),WK2(NP5)                                        EV 00050
      REAL TRACE,SUM                                             EV 00060
      DATA JOB2,IFLAG1,SUM,TRACE/2,0,2*0.0/                      EV 00070
C                                                                EV 00080
C     NP1   :   RAW DATA DIMENSIONALITY                         EV 00090
C     NP3   :   TOTAL NUMBER OF ELEMENTS FOR VCV                EV 00100
C     NP5   :   DIMENSION FOR PERFORMANCE INDEX MATRIX WK2      EV 00110
C                                                                EV 00120
C     XM    :   MEAN VECTOR                                     EV 00130
C     VCV   :   COVARIANCE MATRIX ( SYMMETRIC STORAGE MODE )    EV 00140
C     VCVF  :   COVARIANCE MATRIX ( FULL STORAGE MODE )         EV 00150
C     D     :   EIGENVALUE                                      EV 00160
```

```
C      Z    :   EIGENVECTOR                                            EV 00170
C      WK2  :   PERFORMANCE INDEX MATRIX                               EV 00180
C                                                                      EV 00190
C      11 : INPUT MV FILE ; 12 : OUTPUT EV FILE                        EV 00200
C                                                                      EV 00210
       OPEN(11)                                                        EV 00220
       OPEN(12)                                                        EV 00230
       REWIND 11                                                       EV 00240
       REWIND 12                                                       EV 00250
C                                                                      EV 00260
C      READ IN MEAN VECTOR AND COVARIANCE MATRIX                       EV 00270
C                                                                      EV 00280
       DO 10 I=1,NP1/NF3                                               EV 00290
    10 READ(11,*)(XM(J),J=1+(I-1)*NF3,I*NF3)                           EV 00300
     1 FORMAT(5E15.7)                                                  EV 00310
       DO 20 I=1,NP3/NF3                                               EV 00320
    20 READ(11,*)(VCV(J),J=1+(I-1)*NF3,I*NF3)                          EV 00330
       CALL VCVTSF(VCV,NP1,VCVF,NP1)                                   EV 00340
C                                                                      EV 00350
C      FIND TRACE, EIGENVALUES AND EIGENVECTORS OF THE COVARIANCE MATRIX EV 00360
C                                                                      EV 00370
       DO 30 I=1,NP1                                                   EV 00380
    30 TRACE=TRACE+VCVF(I,I)                                           EV 00390
       CALL EIGRS(VCV,NP1,JOB2,D,Z,NP1,WK2,IER)                        EV 00400
C                                                                      EV 00410
C      PRINT THE PERFORMANCE INDEX AND ACCURACY COMMENTS               EV 00420
C                                                                      EV 00430
       IF(IER.NE.0.OR.WK2(1).GE.1.0)GO TO 40                          EV 00440
       WRITE(*,3)IER,WK2(1)                                            EV 00450
       GO TO 50                                                        EV 00460
    40 WRITE(*,2)IER,WK2(1)                                            EV 00470
     2 FORMAT(' PERFORMANCE OF "EIGRS" IS POOR, IER =',I5,             EV 00480
      +' WK2(1) =',E15.7)                                              EV 00490
     3 FORMAT(' PERFORMANCE OF "EIGRS" IS GOOD, IER =',I5,             EV 00500
      +' WK2(1) =',E15.7)                                              EV 00510
C                                                                      EV 00520
C      INTERNAL CHECKING FOR ACCURACY                                  EV 00530
C                                                                      EV 00540
    50 DO 70 I=1,NP1                                                   EV 00550
       IF(D(I).LE.0.0)GO TO 60                                         EV 00560
       GO TO 70                                                        EV 00570
    60 WRITE(*,4)I,D(I)                                                EV 00580
     4 FORMAT(' EIGEN VALUE IS "< = 0.0" AT I =', I5,                  EV 00590
      +' WHERE D(I) =',E15.7)                                          EV 00600
       IFLAG1=IFLAG1+1                                                 EV 00610
    70 CONTINUE                                                        EV 00620
       IF(IFLAG1.GT.0)GO TO 80                                         EV 00630
       WRITE(*,6)                                                      EV 00640
       GO TO 90                                                        EV 00650
    80 WRITE(*,5)IFLAG1                                                EV 00660
     5 FORMAT(' THERE ARE',I5,' NEGATIVE OR ZERO EIGEN VALUES ')       EV 00670
     6 FORMAT(' ALL EIGEN VALUES ARE GREATER THAN ZERO ')             EV 00680
C                                                                      EV 00690
C      FIND THE SUM OF THE EIGENVALUES AND PRINT THE ACCURACY COMMENTS  EV 00700
C                                                                      EV 00710
    90 CALL VABSMF(D,NP1,1,SUM)                                        EV 00720
       IF(ABS(TRACE-SUM).GT.1.0E-1)GO TO 100                          EV 00730
```

```
      WRITE(*,8)TRACE,SUM                                           EV 00740
      GO TO 110                                                     EV 00750
  100 WRITE(*,7)TRACE,SUM                                           EV 00760
    7 FORMAT(' ACCURACY OF "EIGRS" IS POOR, TRACE =',E15.7,         EV 00770
   +' SUM =',E15.7)                                                 EV 00780
    8 FORMAT(' ACCURACY OF "EIGRS" IS GOOD, TRACE =',E15.7,         EV 00790
   +' SUM =',E15.7)                                                 EV 00800
C                                                                   EV 00810
C     SEND THE RESULTS TO THE OUTPUT DATA FILE                      EV 00820
C                                                                   EV 00830
  110 WRITE(12,9)TRACE,SUM                                          EV 00840
    9 FORMAT(2E15.7)                                                EV 00850
      DO 120 I=1,NP1/NF3                                            EV 00860
  120 WRITE(12,1)(D(NP1+1-J),J=1+(I-1)*NF3,I*NF3)                   EV 00870
      DO 130 J=1,NP1                                                EV 00880
      DO 130 I=1,NP1/NF3                                            EV 00890
  130 WRITE(12,1)(Z(K,NP1+1-J),K=1+(I-1)*NF3,I*NF3)                 EV 00900
      STOP                                                          EV 00910
      END                                                          EV 00920



      PROGRAM NOLBS                                                 BS 00010
      PARAMETER(NP1=100,NTERM=6,NV=50,NZ1=NP1*NV,N1=1,N2=100)       BS 00020
C                                                                   BS 00030
C     FOR FSS VEGETATION DATA : N1 = 1; N2 = 100                    BS 00040
C     FOR SOIL DATA      : N1 = 4; N2 = 192                         BS 00050
C                                                                   BS 00060
C     FOR SOIL DATA ( FROM EFFECTIVE WAVELENGTH 0.52 TO 2.32UM:180 DIM )BS 00070
C     N1=1, N2=180                                                  BS 00080
C                                                                   BS 00090
      REAL X(NP1,NTERM),AVE(NP1),S1(NP1),Z(NP1,NV)                  BS 00100
      DATA Z/NZ1*0.0/                                               BS 00110
C                                                                   BS 00120
C     NP1   :   RAW DATA DIMENSIONALITY                             BS 00130
C     NTERM :   TOTAL NUMBER OF OPTIMAL FUNCTIONS USED IN THE ALGORITHM  BS 00140
C     NV    :   PRESET MAX NUMBER OF N.O.L. BANDS, INCREASE IT IF NEEDED BS 00150
C     N1    :   THE STARTING WAVELENGTH POINT                       BS 00160
C     N2    :   THE ENDING WAVELENGTH POINT                         BS 00170
C                                                                   BS 00180
C     X     :   EIGENVECTOR ( INPUT )                               BS 00190
C     AVE   :   AVERAGE OF THE FIRST 'NTERM' EIGENVECTORS           BS 00200
C     S1    :   SIGNED VERSION OF AVE(NP1)                          BS 00210
C     Z     :   DESIRED N.O.L. BAND FEATURES ( OUTPUT )             BS 00220
C                                                                   BS 00230
C     11  :   INPUT EIGENVECTOR FILE;   12  : OUTPUT N.O.L. BAND FILE  BS 00240
C                                                                   BS 00250
      OPEN(11)                                                      BS 00260
      OPEN(12)                                                      BS 00270
      REWIND 11                                                     BS 00280
      REWIND 12                                                     BS 00290
      READ(11,*)X1,X2                                               BS 00300
      DO 10 I=1,NP1/5                                               BS 00310
   10 READ(11,*)X1,X2,X3,X4,X5                                      BS 00320
C                                                                   BS 00330
```

```
C        READ IN EIGENVECTORS                                    BS 00340
C                                                                BS 00350
         DO 20 ITERM=1,NTERM                                     BS 00360
         DO 20 J=1,NP1/5                                         BS 00370
      20 READ(11,*)(X(I,ITERM),I=1+(J-1)*5,J*5)                  BS 00380
C                                                                BS 00390
C        FIND THE AVERAGE OF THE FIRST 'NTERM' EIGENVECTORS AND  BS 00400
C        ITS SIGNED VERSION                                      BS 00410
C                                                                BS 00420
         DO 40 J=1,NP1                                           BS 00430
         AVE(J)=0.0                                              BS 00440
         DO 30 ITERM=1,NTERM                                     BS 00450
      30 AVE(J)=AVE(J)+X(J,ITERM)/FLOAT(NTERM)                   BS 00460
         IF(NP1.NE.100)GO TO 35                                  BS 00470
         IF(J.GE.45.AND.J.LE.54)AVE(J)=0.0                       BS 00480
         IF(J.GE.70.AND.J.LE.79)AVE(J)=0.0                       BS 00490
      35 IF(AVE(J).LT.0.0)S1(J)=-1.0                             BS 00500
         IF(AVE(J).GT.0.0)S1(J)=1.0                              BS 00510
         IF(AVE(J).EQ.0.0)S1(J)=0.0                              BS 00520
      40 CONTINUE                                                BS 00530
C                                                                BS 00540
C        THE NEXT 3 LINES CAN BE USED TO PLOT AVE(I) AND S1(I)   BS 00550
C                                                                BS 00560
C        DO 50 I=1,NP1                                           BS 00570
C     50 WRITE(12,51)AVE(I),I,S1(I)                              BS 00580
C     51 FORMAT(E15.7,I5,F5.0)                                   BS 00590
         IVEC=1                                                  BS 00600
         Z(N1,IVEC)=ABS(S1(N1))                                  BS 00610
C                                                                BS 00620
C        FIND N.O.L. BAND FEATURES FROM S1                       BS 00630
C                                                                BS 00640
         DO 60 I=N1+1,N2                                         BS 00650
         IF(NP1.NE.100)GO TO 55                                  BS 00660
         IF(I.GE.45.AND.I.LE.54)GO TO 60                         BS 00670
         IF(I.GE.70.AND.I.LE.79)GO TO 60                         BS 00680
      55 IF(S1(I-1).NE.S1(I))IVEC=IVEC+1                         BS 00690
         WRITE(12,*)I,IVEC                                       BS 00700
         IF(IVEC.GE.NV)GO TO 120                                 BS 00710
         Z(I,IVEC)=ABS(S1(I))                                    BS 00720
      60 CONTINUE                                                BS 00730
C                                                                BS 00740
C        NORMALIZE THE FEATURES AND SEND THEM TO THE OUTPUT FILE BS 00750
C                                                                BS 00760
         DO 100 J=1,IVEC                                         BS 00770
         XN1=0.0                                                 BS 00780
         DO 70 I=1,NP1                                           BS 00790
      70 XN1=XN1+Z(I,J)*Z(I,J)                                   BS 00800
         DO 80 I=1,NP1                                           BS 00810
      80 Z(I,J)=Z(I,J)/SQRT(XN1)                                 BS 00820
         DO 90 I1=1,NP1/5                                        BS 00830
      90 WRITE(12,91)(Z(I,J),I=1+(I1-1)*5,I1*5)                  BS 00840
      91 FORMAT(5E15.7)                                          BS 00850
     100 CONTINUE                                                BS 00860
     120 PRINT*,' TOTAL NUMBER OF N.O.L. BAND FEATURES =',IVEC   BS 00870
         STOP                                                    BS 00880
         END                                                     BS 00890
```

```
      PROGRAM WALSH                                                 WAL00010
C                                                                   WAL00020
C     THIS PROGRAM IS USED TO GENERATE THE FIRST 64 100-DIM. WALSH FUN. WAL00030
C     IN THIS PROGRAM WE SET W1=0.1 AND W2=-0.1 SUCH THAT NORM(W)=1.0  WAL00040
C     NP1 = 100, M = 6 , NF4 = 5 USED FOR 64 100-DIM WALSH FUNCTIONS  WAL00050
C                                                                   WAL00060
      PARAMETER(NP1=100,M=6,NTVEC=2**M,NMAX=2**(M-1),               WAL00070
     +W1=0.1,W2=-0.1,NF4=5,NP5=NP1/2,NP6=NP1/4)                     WAL00080
      REAL Z(NP1,NTVEC),ZW1(NP1,NMAX),ZW2(NP1,NMAX)                 WAL00090
      INTEGER NZERO(NTVEC)                                          WAL00100
C                                                                   WAL00110
C     NP1   : DIMENSIONALITY OF WALSH FUNCTION                      WAL00120
C     M     : TOTAL NUMBER OF WALSH FUNCTIONS IS 2**M               WAL00130
C     NTVEC : TOTAL NUMBER OF WALSH FUNCTIONS                       WAL00140
C     W1    : THE NORMALIZED LENGTH OF 100-DIM. WALSH FUNCTION      WAL00150
C     W2    : THE NEGATIVE OF W1                                    WAL00160
C     NF4   : OUTPUT FORMAT USE                                     WAL00170
C     Z     : RESULTS OF WALSH FUNCTIONS ( OUTPUT )                 WAL00180
C     ZW1   : INTERMEDIATE MATRIX FOR WALSH FUNCTION GENERATION     WAL00190
C     ZW2   : INTERMEDIATE MATRIX FOR WALSH FUNCTION GENERATION     WAL00200
C     NZERO : CHECKING VECTOR FOR AXIS CROSSINGS OF WALSH FUNCTIONS WAL00210
C                                                                   WAL00220
C     SET UP THE FIRST 4 WALSH FUNCTIONS                           WAL00230
C                                                                   WAL00240
      DATA ((Z(I,J),I=1,NP1),J=1,4)/NP1*W1,  NP5*W1,NP5*W2,         WAL00250
     +NP6*W1,NP5*W2,NP6*W1,  NP6*W1,NP6*W2,NP6*W1,NP6*W2/           WAL00260
C                                                                   WAL00270
      OPEN(11)                                                      WAL00280
      REWIND 11                                                     WAL00290
C                                                                   WAL00300
C     STORE THE THIRD AND FOURTH WALSH FUNCTIONS                    WAL00310
C                                                                   WAL00320
      DO 10 J=1,2                                                   WAL00330
      DO 10 I=1,NP1                                                 WAL00340
   10 ZW1(I,J)=Z(I,2+J)                                            WAL00350
      PRINT*,'IM = 0,1,2, SEQ : Z(I,1),Z(I,2),ZW1(I,1),ZW1(I,2)'    WAL00360
      DO 20 I=1,NP1                                                 WAL00370
   20 WRITE(*,*)I,Z(I,1),Z(I,2),ZW1(I,1),ZW1(I,2)                  WAL00380
C                                                                   WAL00390
C     GENERATE THE FIRST 2**M WALSH FUNCTIONS                      WAL00400
C                                                                   WAL00410
      DO 70 IM=3,M                                                  WAL00420
      K=2**(IM-1)                                                   WAL00430
      DO 30 IK=1,K-1,2                                              WAL00440
      IKM=(IK+1)/2                                                  WAL00450
      DO 30 I=1,NP5                                                 WAL00460
      ZW2(I,IK)=ZW1(2*I,IKM)                                        WAL00470
   30 ZW2(NP5+I,IK)=((-1.)**(IKM+1))*ZW1(2*I,IKM)                  WAL00480
      DO 40 IK=2,K,2                                                WAL00490
      IKM=IK/2                                                      WAL00500
      DO 40 I=1,NP5                                                 WAL00510
      ZW2(I,IK)=ZW1(2*I,IKM)                                        WAL00520
   40 ZW2(NP5+I,IK)=((-1.)**(IKM))*ZW1(2*I,IKM)                    WAL00530
      DO 50 IK=1,K                                                  WAL00540
      DO 50 I=1,NP1                                                 WAL00550
      Z(I,K+IK)=ZW2(I,IK)                                           WAL00560
   50 ZW1(I,IK)=ZW2(I,IK)                                           WAL00570
```

```
      IF(IM.GE.6)GO TO 70                                           WAL00580
      WRITE(*,1)IM,K                                                WAL00590
    1 FORMAT(' IM = ',I2,', THE SEQ IS ZW2(I,J), J=1,K=',I3)        WAL00600
      DO 60 I=1,NP1                                                 WAL00610
   60 WRITE(*,3)I,(ZW2(I,J),J=1,K)                                  WAL00620
    3 FORMAT(I4,2X,16F4.1)                                          WAL00630
   70 CONTINUE                                                      WAL00640
C                                                                   WAL00650
C     CHECK TOTAL NUMBER OF AXIS CROSSINGS FOR EACH WALSH FUNCTIONS WAL00660
C                                                                   WAL00670
      DO 80 J=1,NTVEC                                               WAL00680
      DO 80 I=1,NP1-1                                               WAL00690
      IF(Z(I,J).NE.Z(I+1,J))NZERO(J)=NZERO(J)+1                     WAL00700
   80 CONTINUE                                                      WAL00710
C                                                                   WAL00720
C     THE FOLLOWING 2 STATEMENTS CAN BE USED FOR INTERNAL CHECKING  WAL00730
C                                                                   WAL00740
C     DO 85 I1=1,NTVEC/8                                            WAL00750
C  85 WRITE(11,86)(NZERO(J),J=1+(I1-1)*8,I1*8)                      WAL00760
   86 FORMAT(8I8)                                                   WAL00770
C     WRITE(*,*)(NZERO(J),J=1,NTVEC)                                WAL00780
      DO 90 J=1,NTVEC                                               WAL00790
      IF(NZERO(J).NE.(J-1))GO TO 200                                WAL00800
   90 CONTINUE                                                      WAL00810
C                                                                   WAL00820
C     SEND THE RESULTS TO OUTPUT FILE                               WAL00830
C                                                                   WAL00840
      DO 140 J=1,NTVEC                                              WAL00850
      DO 140 K=1,NP1/NF4                                            WAL00860
  140 WRITE(11,4)(Z(I,J),I=1+(K-1)*NF4,K*NF4)                       WAL00870
C                                                                   WAL00880
C     CHOOSE FORMAT(10F8.1) IF NF4=10 INSTEAD OF 5                  WAL00890
C                                                                   WAL00900
C   4 FORMAT(10F8.1)                                                WAL00910
    4 FORMAT(5E15.7)                                                WAL00920
  200 STOP                                                          WAL00930
      END                                                           WAL00940




      PROGRAM INFCLIP                                               INF00010
      PARAMETER(NP1=100,NTERM=16,IEV=1)                             INF00020
      REAL X(NP1)                                                   INF00030
C                                                                   INF00040
C     NP1    :  RAW DATA DIMENSIONALITY                             INF00050
C     NTERMS :  TOTAL NUMBER OF OPTIMAL FUNCTIONS USED IN THE ALGORITHMINF00060
C     X      :  INPUT AND OUTPUT VARIABLE                           INF00070
C                                                                   INF00080
C     IEV    :  INPUT FILE READING INDEX ( CHOOSE EITHER 1 OR 0 )   INF00090
C     IEV = 1 IF INPUT FILE CONTAINS TRACE, EIGENVALUES AND THEIR SUM INF00100
C     IEV = 0 IF INPUT FILE CONTAINS ONLY EIGENVECTORS             INF00110
C                                                                   INF00120
C                                                                   INF00130
C     11 :  INPUT EV FILE;  12 :  OUTPUT INF. CLIPPED OPT. FEATURE FILE INF00140
C                                                                   INF00150
```

```
        OPEN(11)                                                    INF00160
        OPEN(12)                                                    INF00170
        REWIND 11                                                   INF00180
        REWIND 12                                                   INF00190
C                                                                   INF00200
C       FIND NORMALIZATION FACTOR                                   INF00210
C                                                                   INF00220
        IF(NP1.EQ.100)XNP1=FLOAT(NP1-20)                            INF00230
        IF(NP1.EQ.200)XNP1=FLOAT(NP1)                               INF00240
C                                                                   INF00250
C       READ INPUT EIGENVECTORS FOR TWO POSSIBLE CASES              INF00260
C                                                                   INF00270
        IF(IEV.EQ.0)GO TO 15                                        INF00280
        READ(11,*)X1,X2                                             INF00290
        DO 10 I=1,NP1/5                                             INF00300
  10    READ(11,*)X1,X2,X3,X4,X5                                    INF00310
C                                                                   INF00320
C       FIND INFINITE CLIPPED VERSION FOR EVERY OPTIMAL FUNCTION    INF00330
C                                                                   INF00340
  15 DO 50 ITERM=1,NTERM                                            INF00350
        DO 20 J=1,NP1/5                                             INF00360
  20 READ(11,*) (X(I),I=1+(J-1)*5,J*5)                              INF00370
        XN1=1./SQRT(XNP1)                                           INF00380
        DO 30 J=1,NP1                                               INF00390
        IF(NP1.EQ.100.AND.J.GE.45.AND.J.LE.54)X(J)=0.0              INF00400
        IF(NP1.EQ.100.AND.J.GE.70.AND.J.LE.79)X(J)=0.0              INF00410
        IF(NP1.EQ.200.AND.J.GE.1.AND.J.LE.3)X(J)=0.0                INF00420
        IF(NP1.EQ.200.AND.J.GE.193.AND.J.LE.200)X(J)=0.0            INF00430
        IF(X(J).GT.0.0)X(J)=XN1                                     INF00440
        IF(X(J).LT.0.0)X(J)=-XN1                                    INF00450
  30 CONTINUE                                                       INF00460
C                                                                   INF00470
C       SEND THE RESULT TO THE OUTPUT FILE                          INF00480
C                                                                   INF00490
        DO 40 J=1,NP1/5                                             INF00500
  40 WRITE(12,41) (X(I),I=1+(J-1)*5,J*5)                            INF00510
  41 FORMAT(5E15.7)                                                 INF00520
  50 CONTINUE                                                       INF00530
        STOP                                                        INF00540
        END                                                         INF00550


        PROGRAM OLBS                                                OLB00010
        PARAMETER(NP1=100,NTERM=6,NV=120,NZ1=2*NV,NZ2=NP1*NV,       OLB00020
       +N1=1,N2=100,W1=0.40,DW=0.02,NVX=40,NV2=NV*NV)               OLB00030
        REAL X(NP1,NTERM),S1(NP1),Z(NP1,NV),T1(NV),                 OLB00040
       +TEST(NP1,NV),A(NP1,NVX)                                     OLB00050
        INTEGER NX(NTERM),NEDGE(2,NV),NWID(NV),NRANK(NV),NREP(NV),  OLB00060
       +MREP(NV)                                                    OLB00070
        DATA NX,Z,NEDGE,NWID/NTERM*0,NZ2*0.0,NZ1*0,NV*0/            OLB00080
        DATA NREP,TEST/NV*1,NZ2*0.0/                                OLB00090
C                                                                   OLB00100
C       NP1    :   RAW DATA DIMENSIONALITY                          OLB00110
```

```
C      NTERM :  TOTAL NUMBER OF OPTIMAL FUNCTIONS USED IN THE ALGORITHM OLB00120
C      NV    :  PRESET TOTAL NUMBER OF L.D. BANDS, INCREASE IT IF NEEDEDOLB00130
C      N1    :  STARTING WAVELENGTH POINT                              OLB00140
C      N2    :  ENDING WAVELENGTH POINT                                OLB00150
C      W1    :  STARTING WAVELENGTH IN MICRO METER ( UM )              OLB00160
C      DW    :  SPECTRAL RESOLUTION ( UM )                             OLB00170
C      NVX   :  PRESET TOTAL NUMBER OF L.I. BANDS, INCREASE IT IF NEEDEDOLB00180
C                                                                      OLB00190
C      X     :  INPUT EIGENVECTOR MATRIX                               OLB00200
C      S1    :  SIGNED VERSION OF THE EIGENVECTOR                      OLB00210
C      Z     :  L.D. BAND FEATURES                                     OLB00220
C      T1    :  TEMPORARY STORAGE VECTOR                               OLB00230
C      TEST  :  OUTPUT O.L. BAND FEATURES ( L.I. FEATURES )            OLB00240
C      A     :  INTERMEDIATE MATRIX FOR RANK TEST                      OLB00250
C      NX(K) :  TOTAL NO. OF L.D. BANDS FOR THE FIRST K EIGENVECTOR(S) OLB00260
C      NEDGE :  BAND EDGES FOR EACH L.D. BANDS                         OLB00270
C      NWID  :  BAND WIDTH FOR EACH L.D. BANDS                         OLB00280
C      NRANK :  POSITIONS OF THE RANKED FEATURES BY THE WIDTHS         OLB00290
C      NREP  :  INDEX SHOWS IF THE L.D. BANDS ARE REPEATED             OLB00300
C      MREP  :  INDEX SHOWS IF THE BANDS ARE L.I. BANDS                OLB00310
C                                                                      OLB00320
C      NREP = 1 IF NON-REPEATED BAND ;  NREP = 0 IF REPEATED           OLB00330
C      MREP = 1 IF L.I. BAND         ;  MREP = 0 IF L.D.               OLB00340
C                                                                      OLB00350
C      11 :  INPUT EIGENVECTOR FILE                                    OLB00360
C      12 :  FIRST OUTPUT FILE  ---- L.D. AND L.I. BAND INFORMATION    OLB00370
C      13 :  SECOND OUTPUT FILE ---- DESIRED O.L. BAND FEATURE         OLB00380
C                                                                      OLB00390
       OPEN(11)                                                        OLB00400
       OPEN(12)                                                        OLB00410
       OPEN(13)                                                        OLB00420
       REWIND 11                                                       OLB00430
       REWIND 12                                                       OLB00440
       REWIND 13                                                       OLB00450
C                                                                      OLB00460
C      READ IN EIGENVECTORS                                            OLB00470
C                                                                      OLB00480
       READ(11,*)X1,X2                                                 OLB00490
       DO 10 I=1,NP1/5                                                 OLB00500
    10 READ(11,*)X1,X2,X3,X4,X5                                        OLB00510
       DO 20 J=1,NTERM                                                 OLB00520
       DO 20 I=1,NP1/5                                                 OLB00530
    20 READ(11,*) (X(K,J),K=1+(I-1)*5,I*5)                             OLB00540
C                                                                      OLB00550
C      FIND THE L.D. BAND FEATURES FROM FIRST 'NTERM' OPTIMAL FUNCTIONS OLB00560
C                                                                      OLB00570
       IVEC=1                                                          OLB00580
       DO 70 J=1,NTERM                                                 OLB00590
       DO 40 I=1,NP1                                                   OLB00600
       IF(X(I,J).LT.0.0)S1(I)=-1.0                                     OLB00610
       IF(X(I,J).GT.0.0)S1(I)=+1.0                                     OLB00620
       IF(X(I,J).EQ.0.0)S1(I)=0.0                                      OLB00630
       IF(NP1.NE.100)GO TO 40                                          OLB00640
       IF(I.GE.45.AND.I.LE.54)S1(I)=0.0                                OLB00650
       IF(I.GE.70.AND.I.LE.79)S1(I)=0.0                                OLB00660
    40 CONTINUE                                                        OLB00670
       Z(N1,IVEC)=ABS(S1(N1))                                          OLB00680
```

```
      DO 60 I=N1+1,N2                                                OLB00690
      IF(NP1.NE.100)GO TO 50                                         OLB00700
      IF(I.GE.45.AND.I.LE.54)GO TO 60                                OLB00710
      IF(I.GE.70.AND.I.LE.79)GO TO 60                                OLB00720
   50 IF(S1(I-1).NE.S1(I))IVEC=IVEC+1                                 OLB00730
      IF(IVEC.GT.NV)GO TO 350                                         OLB00740
      Z(I,IVEC)=ABS(S1(I))                                           OLB00750
   60 CONTINUE                                                        OLB00760
      NX(J)=IVEC                                                      OLB00770
      IVEC=IVEC+1                                                     OLB00780
   70 CONTINUE                                                        OLB00790
C                                                                     OLB00800
C     FIND THE BAND EDGES AND BAND WIDTH FOR EACH L.D. BAND FEATURES  OLB00810
C                                                                     OLB00820
      NVTOT=NX(NTERM)                                                 OLB00830
      DO 90 J=1,NVTOT                                                 OLB00840
      I1=0                                                            OLB00850
      I2=0                                                            OLB00860
      DO 80 I=1,NP1                                                   OLB00870
      CK1=Z(I,J)                                                      OLB00880
      IF(CK1.EQ.0.0)GO TO 80                                          OLB00890
      IF(CK1.NE.0.0.AND.I1.EQ.0)I1=I                                  OLB00900
      IF(CK1.NE.0.0.AND.I1.NE.0)I2=I                                  OLB00910
   80 CONTINUE                                                        OLB00920
      IF(I2.EQ.0)I2=I1                                                OLB00930
      NEDGE(1,J)=I1                                                   OLB00940
      NEDGE(2,J)=I2                                                   OLB00950
      NWID(J)=I2-I1+1                                                 OLB00960
   90 CONTINUE                                                        OLB00970
C                                                                     OLB00980
C     FIND THE WAVELENGTH EDGES AND SEND THEM TO THE FIRST OUTPUT FILE OLB00990
C                                                                     OLB01000
      DO 100 J=1,NTERM                                                OLB01010
      WRITE(12,*)J                                                    OLB01020
      IF(J.EQ.1)NS1=NX(J)                                             OLB01030
      IF(J.NE.1)NS1=NX(J)-NX(J-1)                                     OLB01040
      DO 100 I=1,NS1                                                  OLB01050
      IF(J.EQ.1)NS2=I                                                 OLB01060
      IF(J.NE.1)NS2=I+NX(J-1)                                         OLB01070
      I1=NEDGE(1,NS2)                                                 OLB01080
      I2=NEDGE(2,NS2)                                                 OLB01090
      XW1=W1+FLOAT(I1-1)*DW                                           OLB01100
      XW2=W1+FLOAT(I2)*DW                                             OLB01110
      WRITE(12,101)NS2,I,NEDGE(1,NS2),NEDGE(2,NS2),XW1,XW2,NWID(NS2)  OLB01120
  100 CONTINUE                                                        OLB01130
  101 FORMAT(2I5,2X,I3,1X,'-',I3,2X,'; ',F5.2,1X,'-',F5.2,I5)         OLB01140
      PRINT*,'TOTAL NUMBER OF BANDS IS = ',NVTOT                      OLB01150
C                                                                     OLB01160
C     RANK THE L.D. BAND ACCORDING TO THEIR WIDTHS IN DESCENDING ORDER OLB01170
C     AND SEND THE RESULTS TO THE FIRST OUTPUT FILE                   OLB01180
C                                                                     OLB01190
      DO 110 I=1,NV                                                   OLB01200
  110 T1(I)=FLOAT(NWID(I))                                            OLB01210
      DO 120 I=1,NVTOT                                                OLB01220
      CALL VABMXF(T1(1),NV,1,IMAX,BIG)                                OLB01230
      NRANK(I)=IMAX                                                   OLB01240
      WRITE(12,*)I,NRANK(I),NEDGE(1,IMAX),NEDGE(2,IMAX),NWID(IMAX)    OLB01250
```

```
    120 T1(IMAX)=0.0                                                    OLB01260
C                                                                       OLB01270
C       CHECK IF THE L.D. BAND IS REPEATED. IF IT IS, SET NREP(I) = 0   OLB01280
C                                                                       OLB01290
        DO 140 I=1,NVTOT                                                OLB01300
        DO 130 J=1,NVTOT                                                OLB01310
        IF(I.EQ.J)GO TO 130                                             OLB01320
        I1=NRANK(I)                                                     OLB01330
        I2=NRANK(J)                                                     OLB01340
        I3=NWID(I1)                                                     OLB01350
        I4=NWID(I2)                                                     OLB01360
        IF(I3.NE.I4)GO TO 130                                           OLB01370
        ISTART=NEDGE(1,I1)                                              OLB01380
        JSTART=NEDGE(1,I2)                                              OLB01390
        IEND=NEDGE(2,I1)                                                OLB01400
        JEND=NEDGE(2,I2)                                                OLB01410
        IF(ISTART.EQ.JSTART.AND.IEND.EQ.JEND.AND.I.GT.J)NREP(I)=0       OLB01420
    130 CONTINUE                                                        OLB01430
        IF(NREP(I).EQ.0)GO TO 140                                       OLB01440
        IX=NRANK(I)                                                     OLB01450
C                                                                       OLB01460
C       THE FOLLOWING WRITE STATEMENT CAN BE USED FOR INTERNAL CHECKING OLB01470
C                                                                       OLB01480
C       WRITE(12,131)I,NREP(I),NRANK(I),NEDGE(1,IX),NEDGE(2,IX),NWID(IX)OLB01490
    131 FORMAT(3I4,5X,I4,' -',I4,5X,I4)                                 OLB01500
    140 CONTINUE                                                        OLB01510
C                                                                       OLB01520
C       FIND TOTAL NUMBER OF NON-REPEATED L.D. BAND                     OLB01530
C                                                                       OLB01540
        NDIFF=0                                                         OLB01550
        DO 150 I=1,NVTOT                                                OLB01560
        IF(NREP(I).EQ.1)NDIFF=NDIFF+1                                   OLB01570
    150 MREP(I)=NREP(I)                                                 OLB01580
        PRINT*,'TOTAL NUMBER OF NON-IDENTICAL BANDS IS =',NDIFF         OLB01590
C                                                                       OLB01600
C       FIND L.I. BAND BY CHECKING THE MATRIX RANK                      OLB01610
C                                                                       OLB01620
        ILI=1                                                           OLB01630
        JWID=1                                                          OLB01640
        DO 300 J=1,NVTOT                                                OLB01650
        IF(NREP(J).EQ.0)GO TO 300                                       OLB01660
        JR=NRANK(J)                                                     OLB01670
        DO 160 I=1,NP1                                                  OLB01680
    160 TEST(I,ILI)=Z(I,JR)                                             OLB01690
        DO 170 KI=1,NP1                                                 OLB01700
        DO 170 KJ=1,ILI                                                 OLB01710
    170 A(KI,KJ)=TEST(KI,KJ)                                           OLB01720
C                                                                       OLB01730
C       REDUCE THE MATRIX A TO ITS ECHELON FORM                         OLB01740
C                                                                       OLB01750
        CALL ECHEL(A,NP1,NVX,NP1,ILI)                                   OLB01760
        IEV=0                                                           OLB01770
        DO 190 KI=1,NP1                                                 OLB01780
        DO 180 KJ=1,ILI                                                 OLB01790
        IF(A(KI,KJ).NE.0.0)IEV=IEV+1                                    OLB01800
        IF(A(KI,KJ).NE.0.0)GO TO 190                                    OLB01810
    180 CONTINUE                                                        OLB01820
```

```
      190 CONTINUE                                                     OLB01830
C                                                                      OLB01840
C        SEND THE RANK INFORMATION TO THE FIRST OUTPUT FILE            OLB01850
C        WHERE 'IEV' IS THE RANK AND 'ILI' IS TOTAL NUMBER OF BANDS TESTED OLB01860
C                                                                      OLB01870
         WRITE(12,*)'IEV=',IEV,'; ILI=',ILI,'AT J=',J                  OLB01880
         IF(IEV.LT.ILI)WRITE(12,*)'IEV.LT.ILI AT J=',J                 OLB01890
         IF(IEV.LT.ILI)GO TO 200                                       OLB01900
C                                                                      OLB01910
C        IF RANK IS EQUAL TO TOTAL NO. OF BANDS, TEST THE NEXT WIDEST BAND OLB01920
C                                                                      OLB01930
         IF(IEV.EQ.ILI)ILI=ILI+1                                       OLB01940
         GO TO 300                                                     OLB01950
C                                                                      OLB01960
C        IF RANK IS LESS THEN TOTAL NO. OF BANDS,                      OLB01970
C        ELIMINATE THE WIDEST L.D. BAND                               OLB01980
C                                                                      OLB01990
      200 DO 250 JXLD=1,ILI                                            OLB02000
         DO 210 KJ=1,ILI                                               OLB02010
         DO 210 KI=1,NP1                                               OLB02020
      210 A(KI,KJ)=TEST(KI,KJ)                                         OLB02030
         DO 220 KI=1,NP1                                               OLB02040
      220 A(KI,JXLD)=TEST(KI,ILI)                                      OLB02050
         JLI=ILI-1                                                     OLB02060
         CALL ECHEL(A,NP1,NVX,NP1,JLI)                                 OLB02070
         IEV=0                                                         OLB02080
         DO 240 KI=1,NP1                                               OLB02090
         DO 230 KJ=1,JLI                                               OLB02100
         IF(A(KI,KJ).NE.0.0)IEV=IEV+1                                  OLB02110
         IF(A(KI,KJ).NE.0.0)GO TO 240                                  OLB02120
      230 CONTINUE                                                     OLB02130
      240 CONTINUE                                                     OLB02140
C        PRINT*,'IEV=',IEV,'; ILI=',ILI,'AT J=',J                      OLB02150
C        IF(IEV.LT.ILI)PRINT*,'IEV.LT.ILI AT J=',J                     OLB02160
         IF(IEV.EQ.JLI)J2LD=JXLD                                       OLB02170
         IF(IEV.EQ.JLI)GO TO 260                                       OLB02180
      250 CONTINUE                                                     OLB02190
      260 I1=0                                                         OLB02200
         I2=0                                                          OLB02210
         DO 270 KI=1,NP1                                               OLB02220
         CK1=TEST(KI,J2LD)                                             OLB02230
         IF(CK1.EQ.0.0)GO TO 270                                       OLB02240
         IF(CK1.NE.0.0.AND.I1.EQ.0)I1=KI                               OLB02250
         IF(CK1.NE.0.0.AND.I1.NE.0)I2=KI                               OLB02260
      270 CONTINUE                                                     OLB02270
         IF(I2.EQ.0)I2=I1                                              OLB02280
         DO 275 KI=1,NVTOT                                             OLB02290
         IF(MREP(KI).EQ.0)GO TO 275                                    OLB02300
         MAX=NRANK(KI)                                                 OLB02310
         MEDGE1=NEDGE(1,MAX)                                           OLB02320
         MEDGE2=NEDGE(2,MAX)                                           OLB02330
         IF(I1.EQ.MEDGE1.AND.I2.EQ.MEDGE2)J1LD=KI                      OLB02340
         IF(I1.EQ.MEDGE1.AND.I2.EQ.MEDGE2)GO TO 280                    OLB02350
      275 CONTINUE                                                     OLB02360
      280 MREP(J1LD)=0                                                 OLB02370
C                                                                      OLB02380
C        SEND THE POSITION OF THE WIDEST L.D. BAND FEATURE             OLB02390
```

```
C          TO THE FIRST OUTPUT FILE WHERE :                              OLB02400
C                                                                        OLB02410
C          J1LD IS THE POSITION ON THE VARIABLES NREP AND MREP           OLB02420
C          J2LD IS THE POSITION ON THE RANK CHECKING MATRIX              OLB02430
C                                                                        OLB02440
           WRITE(12,*)'J =',J,'; J1LD =',J1LD,'; J2LD =',J2LD            OLB02450
           DO 290 J1=J2LD,ILI-1                                          OLB02460
           DO 290 I1=1,NP1                                               OLB02470
   290 TEST(I1,J1)=TEST(I1,J1+1)                                         OLB02480
   300 CONTINUE                                                          OLB02490
C                                                                        OLB02500
C          SEND THE L.I. INDEX TO THE FIRST OUTPUT FILE                  OLB02510
C                                                                        OLB02520
           PRINT*,'TOTAL NUMBER OF L.I. BANDS IS =',IEV                  OLB02530
           DO 310 I=1,NVTOT                                              OLB02540
   310 WRITE(12,*)I,NREP(I),MREP(I)                                      OLB02550
C                                                                        OLB02560
C          NORMALIZE THE O.L. BANDS AND SEND THEM TO THE SECOND OUTPUT FILE OLB02570
C                                                                        OLB02580
           DO 330 J=1,IEV                                                OLB02590
           XN1=0.0                                                       OLB02600
           DO 320 I=1,NP1                                                OLB02610
           IF(TEST(I,J).EQ.1)XN1=XN1+1                                   OLB02620
   320 CONTINUE                                                          OLB02630
           DO 330 I=1,NP1                                                OLB02640
   330 TEST(I,J)=TEST(I,J)/SQRT(XN1)                                     OLB02650
           DO 340 J=1,IEV                                                OLB02660
           DO 340 K=1,NP1/5                                              OLB02670
C          J1=IEV-J+1                                                    OLB02680
   340 WRITE(13,341)(TEST(I,J),I=1+(K-1)*5,K*5)                          OLB02690
   341 FORMAT(5E15.7)                                                    OLB02700
           GO TO 360                                                     OLB02710
   350 PRINT*,'TOTAL NUMBER OF BANDS IS OUT OF PRESET LIMIT'             OLB02720
   360 STOP                                                              OLB02730
           END                                                           OLB02740
           SUBROUTINE ECHEL(A,NP1,NVX,NROW,NCOL)                         OLB02750
           REAL A(NP1,NVX)                                               OLB02760
C                                                                        OLB02770
C          THIS SUBROUTINE REDUCES MATRIX A INTO ITS ECHELON FORM        OLB02780
C                                                                        OLB02790
           JCOL=1                                                        OLB02800
           IROW=1                                                        OLB02810
     5 DO 100 I=IROW,NROW                                                OLB02820
           IF(A(I,JCOL).EQ.0.0)GO TO 100                                 OLB02830
C          INTERCHANGE I AND IROW TO GET NONZERO PIVOT                   OLB02840
           IF(I.EQ.IROW)GO TO 20                                         OLB02850
           DO 10 J=JCOL,NCOL                                             OLB02860
           X1=A(I,J)                                                     OLB02870
           A(I,J)=A(IROW,J)                                              OLB02880
    10 A(IROW,J)=X1                                                      OLB02890
C          NORMALIZE ROW TO GET POSITIVE NUMBER FOR PIVOT                OLB02900
    20 IF(A(IROW,JCOL).GT.0.0)GO TO 40                                   OLB02910
           DO 30 J=JCOL,NCOL                                             OLB02920
    30 A(IROW,J)=-A(IROW,J)                                              OLB02930
    40 IF(IROW.GE.NROW)RETURN                                           OLB02940
C          ZERO COLUMN BELOW PIVOT                                       OLB02950
           IROWX=IROW+1                                                  OLB02960
```

```
      DO 60 K=IROWX,NROW                                          OLB02970
      X1=A(K,JCOL)                                                OLB02980
      IF(X1.EQ.0.0)GO TO 60                                       OLB02990
      DO 50 J=JCOL,NCOL                                           OLB03000
   50 A(K,J)=-X1*A(IROW,J)+A(K,J)                                 OLB03010
   60 CONTINUE                                                    OLB03020
      IROW=IROW+1                                                 OLB03030
      JCOL=JCOL+1                                                 OLB03040
      GO TO 5                                                     OLB03050
  100 CONTINUE                                                    OLB03060
      IF(IROW.GT.NROW)RETURN                                      OLB03070
      JCOL=JCOL+1                                                 OLB03080
      GO TO 5                                                     OLB03090
      END                                                         OLB03100




      PROGRAM CLST                                                CLS00010
      PARAMETER(NTERM=16,MTERM=NTERM*(NTERM+1)/2,NCLS=3,NP1=100,  CLS00020
     +NSET=1,MSET=1,NDSET=1,NTSET=1,NF2=10,NF3=5,NSMAX=1000,      CLS00030
     +NKLT=0,IEV=0,NLI=16,VLD=-0.0,NSAMP=10,NF=NF2)              CLS00040
C                                                                 CLS00050
C     NKLT = 1  :     JUST FIND TRANSFORMED DATA XKLT             CLS00060
C     NKLT = 0  :     FIND XKLT AND CLASS STATISTICS              CLS00070
C                                                                 CLS00080
C     NF = NF2 = 10 USED TO READ (10F8.3) RAW DATA                CLS00090
C     NF = NF3 =  5 USED TO READ (5E15.7) CANONICAL TRANSFORMED DATA CLS00100
C     WHEN :  NF = NF3 = 5 --> NP1 MUST BE REDUCED TO LOWER DIM.  CLS00110
C                                                                 CLS00120
C     NTERM = TOTAL NUMBER OF FEATURES (MAY NOT ALL BE NUMERICALLY L.I.)CLS00130
C     NCLS  = TOTOAL NUMBER OF INFORMATION CLASSES                CLS00140
C     NP1   = DIMENSIONALITY OF INPUT DATA                        CLS00150
C             NP1 = RAW DATA DIMENSIONALTY IF USED IN DATA PREPROCESSINGCLS00160
C             NP1 = TRANSFORMED DATA DIMENSIONALTY IF USED IN CAN. ANAL.CLS00170
C     IEV   = INPUT FEATURE READING INDEX, EITHER 1 OR 0          CLS00180
C             IEV = 0 IF FEATURE FILE DOES NOT CONTAIN TRACE & EVALUES CLS00190
C             IEV = 1 IF FEATURE FILE CONTAINS TRACE & EVALUES    CLS00200
C     NLI   = TOTAL NUMBER OF L.I. FEATURES DESIRED               CLS00210
C     NSMAX = PRESET MAX. NO. OF SAMPLES FOR ONE CLASS            CLS00220
C     NSAMP = TOTAL NUMBER OF TEST SAMPLES USED TO CHECK POS. DEF. CLS00230
C                                                                 CLS00240
C                                                                 CLS00250
      REAL X(NSMAX,NTERM),Z(NP1,NTERM),RX(NP1),                   CLS00260
     +T1(NP1),T2(NP1),T3(NP1),XT(NP1),XM(NP1),D(NP1),             CLS00270
     +XMCT(NTERM,NCLS),XMC(NTERM),W(NP1),T(NP1),                  CLS00280
     +VCT(MTERM,NCLS),VC(MTERM),CT(NCLS),                         CLS00290
     +VCIT(MTERM,NCLS),VCI(MTERM),TEST(NTERM,NTERM),              CLS00300
     +VCIF(NTERM,NTERM),VCF(NTERM,NTERM),                         CLS00310
     +VCTF(NTERM,NTERM,NCLS),XMCTF(NTERM,NCLS),                   CLS00320
     +VCTLI(MTERM,NCLS),XMTLI(NTERM,NCLS),                        CLS00330
     +WK(NTERM),VCV(MTERM),VEC(NSAMP,NTERM)                       CLS00340
      INTEGER NBR(6),NST(NCLS,NTSET)                              CLS00350
      DOUBLE PRECISION DSEED                                      CLS00360
      DATA (NBR(I),I=4,6),W/1,0,0,NP1*1.0/                        CLS00370
C                                                                 CLS00380
```

```
C     X     = TRANSFORMED DATA                                         CLS00390
C     Z     = FEATURES                                                 CLS00400
C     RX    = TEMPORARY STORAGE FOR FEATURES                           CLS00410
C     XT    = INPUT DATA                                               CLS00420
C     XM    = MEAN VECTOR                                              CLS00430
C     D     = EIGENVALUES                                              CLS00440
C     XMCT  = MEAN VECTOR FOR ALL CLASSES                              CLS00450
C     XMC   = MEAN VECTOR FOR ONE CLASS                                CLS00460
C     VCT   = COVARIANCE MATRIX FOR ALL CLASSES                        CLS00470
C     VC    = COVARIANCE MATRIX FOR ONE CLASS                          CLS00480
C     VCIT  = INVERSE MATRIX OF ALL CLASS COVARIANCE MATRICES          CLS00490
C     VCI   = INVERSE MATRIX OF ONE CLASS COVARIANCE MATRIX            CLS00500
C     TEST  = INTERNAL MATRIX INVERSION CHECKING MATRIX                CLS00510
C     VCTLI = COV. MATRIX FOR ALL CLASSES BY USING ALL L.I. FEATURES   CLS00520
C     XMTLI = MEAN VECTOR FOR ALL CLASSES BY USING L.I. FEATURES       CLS00530
C     WK    = WORKING SPACE FOR IMSL ROUTINES                          CLS00540
C     VCV   = COV. MATRIX USED TO TEST ITS POSITIVE DEFINITENESS       CLS00550
C     VEC   = GENERATED SAMPLES USED TO TEST POSITIVE DEFINITENESS     CLS00560
C                                                                      CLS00570
C--->>CHOOSE OR TYPE IN THE CORRECT NUMBERS OF SAMPLES IN THE DATA SETS CLS00580
C                                                                      CLS00590
C -----------------------------------------------------------------CLS00600
C    NSET    F1    NP2    A       B       C     DACO    EXNU    RUSE   CLS00610
C     1   M2611K1   832  WW:141 SF:414 GS:277 760928 76102207    1-1622 CLS00620
C     2   M2611K2  1551  WW:658 SF:211 UC:682 770503 77102207 6515-8096 CLS00630
C     3   M2611K3  1477  WW:677 SF:643 GS:157 770626 77102207 8097-9691 CLS00640
C     4   M2614N1  1265  SW:664 SF:437 NP:164 770508 77102217    1-1396 CLS00650
C     5   M2614N2  1239  SW:787 SF:291 NP:161 770629 77102217 2777-4141 CLS00660
C     6   M2614N3  1444  SW:931 SF:330 NP:183 770804 77102217 5426-6993 CLS00670
C                                                                      CLS00680
C    DATA NST/141,414,277,658,211,682,677,643,157/                    CLS00690
C    DATA NST/141,414,277,658,211,682,677,643,157,                    CLS00700
C   +664,437,164,787,291,161,931,330,183/                             CLS00710
C    DATA NST/664,437,164,787,291,161,931,330,183/                    CLS00720
C    DATA NST/141,414,277,658,211,682,677/                            CLS00730
C    DATA NST/587,216,121/                                            CLS00740
C    DATA NST/658,211,682/                                            CLS00750
C                                                                      CLS00760
C ------------------------------------------------------------------CLS00770
C    THE FOLLOWING DATA 'NST' ARE USED FOR SOIL ORDER DATA SET. 'SO'  CLS00780
C                                                                      CLS00790
C    NP2=479; MOL ALF EN AR UL IN SP VE H OX UNCLASSIFIED             CLS00800
C    DATA NST/154,113,78,52,45,37,30,11,8,11,32/                      CLS00810
C    DATA NST/154,113,78,52,45,97/                                    CLS00820
C    DATA NST/154,113,78,52,45,37/                                    CLS00830
C                                                                      CLS00840
C ------------------------------------------------------------------CLS00850
C                                                                      CLS00860
C    THE FOLLOWING DATA 'NST' IS USED FOR SOIL 'OM1' DATA SET         CLS00870
C    I.E. (1) MOLLISOL, OR (2) ALFISOL, AND GROUP SAMPLES             CLS00880
C    ACCORDING TO THEIR ORGANIC MATERIAL: % WEIGHT                    CLS00890
C    CLASS 1 TO 6 :  NP2 = 255                                        CLS00900
C    CLS1 : .11% .GE. OM .LE. 1.5% :  #  1 -> # 51                    CLS00910
C    CLS2 : 1.5% .GT. OM .LE. 2.0% :  # 52 -> # 104                   CLS00920
C    CLS3 : 2.0% .GT. OM .LE. 2.5% :  # 105 -> # 138                  CLS00930
C    CLS4 : 2.5% .GT. OM .LE. 3.5% :  # 139 -> # 183                  CLS00940
C    CLS5 : 3.5% .GT. OM .LE. 5.0% :  # 184 -> # 222                  CLS00950
```

```
C      CLS6 : 5.0% .GT. OM .LE. 10.12% : # 223 -> # 255         CLS00960
C                                                               CLS00970
C      DATA NST/51,54,33,45,39,33/                              CLS00980
C                                                               CLS00990
C      DATA 'S2A' :  ANOTHER TEST GROUPED BY THE SAME OM RANGES AS 'OM2' CLS01000
C      OM PERCENTAGE :  0,1; 1,2; 2,3; 3,4; 4,6; 6 AND ABOVE    CLS01010
C                                                               CLS01020
C      DATA NST/26,78,64,32,55/                                 CLS01030
C                                                               CLS01040
C                                                               CLS01050
C      ------------------------------------------------------------     CLS01060
C      THE FOLLOWING DATA 'NST' IS USED FOR 'OM2' DATA SET      CLS01070
C      ACCORDING TO THEIR ORGANIC MATERIAL: % WEIGHT            CLS01080
C      CLASS 1 TO 6 :  NP2 = 514                                CLS01090
C      CLS1 : .08% .GE. OM .LE. 1.0%  : #  1 -> # 82            CLS01100
C      CLS2 : 1.0% .GT. OM .LE. 2.0%  : # 83 -> # 217           CLS01110
C      CLS3 : 2.0% .GT. OM .LE. 3.0%  : # 218 -> # 337          CLS01120
C      CLS4 : 3.0% .GT. OM .LE. 4.0%  : # 338 -> # 391          CLS01130
C      CLS5 : 4.0% .GT. OM .LE. 6.0%  : # 392 -> # 450          CLS01140
C      CLS6 : 6.0% .GT. OM .LE. 84.79% : # 451 -> # 514         CLS01150
C                                                               CLS01160
C      DATA NST/82,135,120,54,59,64/                            CLS01170
C                                                               CLS01180
C      DATA NST/82,135,120,54,123/                              CLS01190
C      DATA NST/44,31,18,23,24,51,37,27/                        CLS01200
C      DATA NST/83,57,94,31,37,59,103,26,24/                    CLS01210
C      DATA NST/103,26,24/                                      CLS01220
C      ------------------------------------------------------------     CLS01230
C      THE FOLLOWING DATA 'NST' IS USED FOR SOIL IRON OXIDE 'IO' DATA SETCLS01240
C      ACCORDING TO THEIR FE2O3 % WEIGHT                        CLS01250
C      CLASS 1 TO 6 :  NP2 = 467                                CLS01260
C      CLS1 : .02% .GE. FE2O3 .LE. 0.4%  : #  1 -> # 102        CLS01270
C      CLS2 : 0.4% .GT. FE2O3 .LE. 0.6%  : # 103 -> # 175       CLS01280
C      CLS3 : 0.6% .GT. FE2O3 .LE. 0.8%  : # 176 -> # 244       CLS01290
C      CLS4 : 0.8% .GT. FE2O3 .LE. 1.2%  : # 245 -> # 349       CLS01300
C      CLS5 : 1.2% .GT. FE2O3 .LE. 1.6%  : # 350 -> # 401       CLS01310
C      CLS6 : 1.6% .GT. FE2O3 .LE. 25.60% : # 402 -> # 467      CLS01320
C                                                               CLS01330
C      DATA NST/102,73,69,105,52,66/                            CLS01340
C                                                               CLS01350
C      ------------------------------------------------------------     CLS01360
C      THE FOLLOWING DATA 'NST' IS USED FOR SOIL TEXTURE 'ST' DATA SET   CLS01370
C      ACCORDING TO THEIR SAND-SILT-CLAY % CONTENT              CLS01380
C      CLASS 1 TO 6 :  NP2 = 483; DETAILS : SEE FILE ( S5L.DATA.C1)  CLS01390
C                                                               CLS01400
C      DATA NST/40,63,76,93,68,143/                             CLS01410
C                                                               CLS01420
C      ------------------------------------------------------------     CLS01430
C      THE FOLLOWING DATA 'NST' IS USED FOR S.D. VEGETATION DATA  CLS01440
C                                                               CLS01450
C      DATA NST/225,61,292,469, 82,182,63,103, 39,39,217,51,    CLS01460
C      +393,441,80,88, 88,41,32,26, 118,43,121,44, 45,102,66,89, CLS01470
C      +78,53,147,39, 24,42,119,69, 76,96,107,154, 28,19/       CLS01480
C                                                               CLS01490
C      ------------------------------------------------------------     CLS01500
C      THE FOLLOWING DATA 'NST' IS USED FOR IOWA VEGETATION DATA  CLS01510
C                                                               CLS01520
```

```
C        DATA NST/514,41, 517,36,32, 621,517,45, 610,485,21,         CLS01530
C       +437,377,22, 190,172,25, 650,568,42, 435,417,44, 393,267/    CLS01540
C                                                                     CLS01550
C       ------------------------------------------------------------  CLS01560
C                                                                     CLS01570
C       11 = DATA;   12 = FEATURES;   13 = CLASS STATISTICS;          CLS01580
C       14 = TRANSFORMED DATA  ; 15 = LDBAND ;    16 = RANDOM         CLS01590
C                                                                     CLS01600
        OPEN(11)                                                      CLS01610
        OPEN(12)                                                      CLS01620
        OPEN(13)                                                      CLS01630
        OPEN(14)                                                      CLS01640
        OPEN(15)                                                      CLS01650
        REWIND 11                                                     CLS01660
        REWIND 12                                                     CLS01670
        REWIND 13                                                     CLS01680
        REWIND 14                                                     CLS01690
        REWIND 15                                                     CLS01700
C                                                                     CLS01710
C       SET UP DATA INPUT&OUTPUT DO LOOP PARAMETERS                   CLS01720
C                                                                     CLS01730
        IK1=MOD(NCLS,6)                                               CLS01740
        IM1=6*(NCLS/6)+1                                              CLS01750
        ILP1=NCLS/6                                                   CLS01760
        IF(ILP1.EQ.0)ILP1=1                                           CLS01770
        IK2=MOD(NTERM,5)                                              CLS01780
        IM2=5*(NTERM/5)+1                                             CLS01790
        ILP2=NTERM/5                                                  CLS01800
        IF(ILP2.EQ.0)ILP2=1                                           CLS01810
        DO 650 ISET=NSET,MSET,NDSET                                   CLS01820
C                                                                     CLS01830
C       READ FEATURE FILE IN TWO CASES ( IEV = 0 OR 1 )              CLS01840
C                                                                     CLS01850
        IF(IEV.EQ.0)GO TO 10                                          CLS01860
        READ(12,*)TRACE,SUM                                           CLS01870
        CALL SR1(12,NP1,NF3,D)                                        CLS01880
   10   DO 30 JTERM=1,NTERM                                           CLS01890
        CALL SR1(12,NP1,NF3,RX)                                       CLS01900
        DO 20 I=1,NP1                                                 CLS01910
   20   Z(I,JTERM)=RX(I)                                              CLS01920
   30   CONTINUE                                                      CLS01930
C                                                                     CLS01940
C       FIND MEAN VECTOR AND COVARIANCE MATRIX FOR EACH CLASS        CLS01950
C       IN THE FEATURE TRANSFORMED DATA                              CLS01960
C                                                                     CLS01970
        DO 150 LTERM=NTERM,NTERM                                      CLS01980
        KTERM=LTERM*(LTERM+1)/2                                       CLS01990
        DO 150 ICLS=1,NCLS                                            CLS02000
        NS=NST(ICLS,ISET)                                             CLS02010
        PRINT*,' ISET =',ISET,';', LTERM,ICLS,NS                      CLS02020
        DO 40 I=1,NSMAX                                               CLS02030
        DO 40 J=1,NTERM                                               CLS02040
   40   X(I,J)=0.0                                                    CLS02050
        DO 100 ISAMP=1,NS                                             CLS02060
        CALL SR1(11,NP1,NF,XT)                                        CLS02070
        DO 70 JTERM=1,LTERM                                           CLS02080
        DO 60 I=1,NP1                                                 CLS02090
```

```
      T1(I)=XT(I)                                              CLS02100
      T2(I)=W(I)*T1(I)                                         CLS02110
   60 T3(I)=Z(I,JTERM)                                         CLS02120
      CALL VIPRFF(T3,T2,NP1,1,1,XIP)                           CLS02130
   70 X(ISAMP,JTERM)=XIP                                       CLS02140
C                                                              CLS02150
C     SEND THE RESULTS TO THE TRANSFORMED DATA FILE            CLS02160
C                                                              CLS02170
      IF(NTERM.LT.5)GO TO 90                                   CLS02180
      DO 80 I1=1,ILP2                                          CLS02190
   80 WRITE(14,91)(X(ISAMP,J1),J1=1+(I1-1)*5,I1*5)             CLS02200
      IF(IK2.EQ.0)GO TO 100                                    CLS02210
   90 WRITE(14,91)(X(ISAMP,J1),J1=IM2,NTERM)                   CLS02220
   91 FORMAT(5E15.7)                                           CLS02230
  100 CONTINUE                                                 CLS02240
C                                                              CLS02250
C     FIND THE CLASS STATISTICS IF NKLT = 0                    CLS02260
C                                                              CLS02270
      IF(NKLT.EQ.1)GO TO 150                                   CLS02280
      NBR(1)=LTERM                                             CLS02290
      NBR(2)=NS                                                CLS02300
      NBR(3)=NS                                                CLS02310
      DO 110 I=1,NP1                                           CLS02320
  110 T(I)=0.0                                                 CLS02330
      CALL BECOVM(X,NSMAX,NBR,T,XMC,VC,IER)                    CLS02340
C                                                              CLS02350
C     STORE THE CLASS STATISTICS FOR POSITIVE DEFINITENESS CHECKING CLS02360
C                                                              CLS02370
      DO 120 I=1,LTERM                                         CLS02380
C     WRITE(*,*)LTERM,I,XMC(I)                                 CLS02390
  120 XMCT(I,ICLS)=XMC(I)                                      CLS02400
      DO 130 I=1,KTERM                                         CLS02410
C     WRITE(*,*)LTERM,I,VC(I)                                  CLS02420
  130 VCT(I,ICLS)=VC(I)                                        CLS02430
      PRINT*,' THE IER MUST BE "0" FOR BECOVM '                CLS02440
      PRINT*,IER                                               CLS02450
  150 CONTINUE                                                 CLS02460
C                                                              CLS02470
C     STOP THE PROGRAM IF ONLY WANT TO FIND TRANSFORMED DATA (NKLT=1) CLS02480
C                                                              CLS02490
      IF(NKLT.EQ.1)GO TO 650                                   CLS02500
C                                                              CLS02510
C     STORE THE CLASS STATISTICS INTO FULL STORAGE MODE FOR CHECKING CLS02520
C                                                              CLS02530
      DO 170 ICLS=1,NCLS                                       CLS02540
      DO 170 I=1,NTERM                                         CLS02550
      DO 160 J=1,I                                             CLS02560
      IND=I*(I-1)/2+J                                          CLS02570
      VCTF(I,J,ICLS)=VCT(IND,ICLS)                             CLS02580
      VCTF(J,I,ICLS)=VCTF(I,J,ICLS)                            CLS02590
C     WRITE(*,*)I,J,IND,VCTF(I,J,ICLS),VCTF(J,I,ICLS)          CLS02600
  160 CONTINUE                                                 CLS02610
      XMCTF(I,ICLS)=XMCT(I,ICLS)                               CLS02620
  170 CONTINUE                                                 CLS02630
C                                                              CLS02640
C     START CHECKING THE POSITIVE DEFINITENESS OF THE COV. MATRICES CLS02650
C     IF 'LTERM'TH FEATURE IS L.D. ON THE OTHER FEATURES,   THE RELATED CLS02660
```

```
C         ELEMENTS IN THE MEAN VECTORS AND COVARIANCES WILL BE REMOVED    CLS02670
C                                                                         CLS02680
                                                                          CLS02690
          ILI=1                                                           CLS02700
          JLI=ILI*(ILI+1)/2                                               CLS02710
          DO 600 LTERM=1,NTERM                                            CLS02720
          KTERM=LTERM*(LTERM+1)/2                                         CLS02730
          DO 400 ICLS=1,NCLS                                              CLS02740
          IX=0                                                            CLS02750
          DO 200 IROW=1,LTERM                                             CLS02760
          V1=0.0                                                          CLS02770
          DO 180 JCK=1,LTERM                                              CLS02780
      180 V1=V1+VCTF(IROW,JCK,ICLS)                                       CLS02790
          VCK=VLD*LTERM                                                   CLS02800
          IF(V1.EQ.VCK)GO TO 200                                          CLS02810
          IX=IX+1                                                         CLS02820
          IY=0                                                            CLS02830
          DO 190 JCOL=1,LTERM                                             CLS02840
          V2=VCTF(IROW,JCOL,ICLS)                                         CLS02850
          IF(V2.EQ.VLD)GO TO 190                                          CLS02860
          IY=IY+1                                                         CLS02870
          VCF(IX,IY)=V2                                                   CLS02880
      190 CONTINUE                                                        CLS02890
      200 CONTINUE                                                        CLS02900
C         WRITE(15,*)IX,IY,ILI                                            CLS02910
C         PRINT*,'IX,IY,ILI MUST BE THE SAME',IX,IY,ILI                   CLS02920
          CALL VCVTFS(VCF,ILI,NTERM,VC)                                   CLS02930
C         WRITE(*,*)ICLS,VC(1)                                            CLS02940
          OPEN(16)                                                        CLS02950
          REWIND 16                                                       CLS02960
          DO 210 I=1,JLI                                                  CLS02970
          WRITE(16,211)VC(I)                                             CLS02980
C         VCV(I)=VC(I)                                                    CLS02990
      210 VCTLI(I,ICLS)=VC(I)                                             CLS03000
      211 FORMAT(E13.5)                                                   CLS03010
          OPEN(16)                                                        CLS03020
          REWIND 16                                                       CLS03030
          DO 220 I=1,JLI                                                  CLS03040
      220 READ(16,211)VCV(I)                                              CLS03050
          DO 230 I=1,NTERM                                                CLS03060
      230 WK(I)=0.0                                                       CLS03070
          DSEED=5.D0                                                      CLS03080
C                                                                         CLS03090
C         SECOND TEST ON NUMERICAL POSITIVE DEFINITENESS OF THE MATRICES  CLS03100
C                                                                         CLS03110
          CALL GGNSM(DSEED,NSAMP,ILI,VCV,NSAMP,VEC,WK,IER)                CLS03110
          IF(IER.NE.0)GO TO 440                                           CLS03120
C         WRITE(*,*)ICLS,VCTLI(1,ICLS),VC(1)                             CLS03130
C                                                                         CLS03140
C         CHECK IF ALL CLASS COVARIANCES HAVE INVERSE MATRICES            CLS03150
C         VC WILL BE CHANGED AFTER LINV1P                                 CLS03160
C                                                                         CLS03170
          CALL LINV1P(VC,ILI,VCI,IDGT,D1,D2,IER)                          CLS03180
C         WRITE(*,*)ICLS,VCI(1)                                           CLS03190
C         PRINT*,' THE FOLLOWING IER MUST BE 0 FOR LINV1P'                CLS03200
C         PRINT*,ISET,LTERM,ICLS,'; IER =',IER                           CLS03210
          IF(IER.NE.0)GO TO 450                                           CLS03220
          DO 240 I=1,JLI                                                  CLS03230
```

```
    240 VCIT(I,ICLS)=VCI(I)                                           CLS03240
C                                                                     CLS03250
C       STORE BACK THE VALUES OF VCVC FROM VCVCF                      CLS03260
C                                                                     CLS03270
        CALL VCVTFS(VCF,ILI,NTERM,VC)                                 CLS03280
        CALL VCVTSF(VCI,ILI,VCIF,NTERM)                               CLS03290
        DET=D1*2.**D2                                                 CLS03300
        CX=(2.*3.14159)**(FLOAT(ILI)/2.)                             CLS03310
        C=1./(CX*SQRT(DET))                                           CLS03320
        CT(ICLS)=C                                                    CLS03330
        IF(ICLS.NE.NCLS)GO TO 400                                     CLS03340
C                                                                     CLS03350
C       SEND THE FINAL RESULTS TO THE CLASS STATISTICS FILE           CLS03360
C                                                                     CLS03370
        DO 250 KCLS=1,NCLS                                            CLS03380
        IX=0                                                          CLS03390
        DO 250 I=1,LTERM                                              CLS03400
        V3=XMCTF(I,KCLS)                                              CLS03410
        IF(V3.EQ.VLD)GO TO 250                                        CLS03420
        IX=IX+1                                                       CLS03430
        XMTLI(IX,KCLS)=V3                                             CLS03440
    250 CONTINUE                                                      CLS03450
        DO 280 I=1,ILI                                                CLS03460
        IF(NCLS.LT.6)GO TO 270                                        CLS03470
        DO 260 IL=1,ILP1                                              CLS03480
    260 WRITE(13,321)(XMTLI(I,LCLS),LCLS=1+(IL-1)*6,IL*6)            CLS03490
        IF(IK1.EQ.0)GO TO 280                                         CLS03500
    270 WRITE(13,321)(XMTLI(I,LCLS),LCLS=IM1,NCLS)                    CLS03510
    280 CONTINUE                                                      CLS03520
        DO 310 I=1,JLI                                                CLS03530
        IF(NCLS.LT.6)GO TO 300                                        CLS03540
        DO 290 IL=1,ILP1                                              CLS03550
    290 WRITE(13,321)(VCTLI(I,LCLS),LCLS=1+(IL-1)*6,IL*6)            CLS03560
        IF(IK1.EQ.0)GO TO 310                                         CLS03570
    300 WRITE(13,321)(VCTLI(I,LCLS),LCLS=IM1,NCLS)                    CLS03580
    310 CONTINUE                                                      CLS03590
        IF(NCLS.LT.6)GO TO 330                                        CLS03600
        DO 320 IL=1,ILP1                                              CLS03610
    320 WRITE(13,321)(CT(LCLS),LCLS=1+(IL-1)*6,IL*6)                 CLS03620
    321 FORMAT(6E13.5)                                                CLS03630
        IF(IK1.EQ.0)GO TO 340                                         CLS03640
    330 WRITE(13,321)(CT(LCLS),LCLS=IM1,NCLS)                         CLS03650
    340 DO 370 I=1,JLI                                                CLS03660
        IF(NCLS.LT.6)GO TO 360                                        CLS03670
        DO 350 IL=1,ILP1                                              CLS03680
    350 WRITE(13,321)(VCIT(I,LCLS),LCLS=1+(IL-1)*6,IL*6)             CLS03690
        IF(IK1.EQ.0)GO TO 370                                         CLS03700
    360 WRITE(13,321)(VCIT(I,LCLS),LCLS=IM1,NCLS)                     CLS03710
    370 CONTINUE                                                      CLS03720
    400 CONTINUE                                                      CLS03730
C                                                                     CLS03740
C       INTERNAL CHECKING FOR ACCURACY OF MATRIX INVERSION            CLS03750
C                                                                     CLS03760
        DO 430 ICLS=1,NCLS                                            CLS03770
        DO 410 I=1,JLI                                                CLS03780
        VC(I)=VCTLI(I,ICLS)                                           CLS03790
    410 VCI(I)=VCIT(I,ICLS)                                           CLS03800
```

```
C          DO 420 I=1,JLI                                               CLS03810
C      420 WRITE(*,*)VC(I),VCI(I)                                       CLS03820
          CALL VMULSS(VC,VCI,ILI,TEST,NTERM)                           CLS03830
C                                                                       CLS03840
C          THE FOLLOWING 3 STATEMENTS CAN BE USED FOR MATRIX INVERSION CHECK CLS03850
C                                                                       CLS03860
C          PRINT*,' THE FOLLOWING MATRIX MUST BE AN IDENTITY MATRIX '   CLS03870
C          DO 430 I=1,ILI                                               CLS03880
C          WRITE(*,421)(TEST(I,J),J=1,ILI)                             CLS03890
      421 FORMAT(16F5.2)                                                CLS03900
      430 CONTINUE                                                      CLS03910
          PRINT*,' ILI =',ILI                                          CLS03920
          ILI=ILI+1                                                     CLS03930
          JLI=ILI*(ILI+1)/2                                            CLS03940
          IF(ILI.GT.NLI)GO TO 650                                      CLS03950
          GO TO 600                                                     CLS03960
C                                                                       CLS03970
C          SEND THE INFORMATION OF L.D. FEATURES & REASONS FOR          CLS03980
C          NON-POSITIVE-DEFINITENESS OF COV. MATRICES TO THE FILE 'LDBAND' CLS03990
C                                                                       CLS04000
      440 WRITE(15,*)'GGNSM HAS IER.NE.0'                              CLS04010
      450 WRITE(15,*)'ISET =',ISET,';LTERM =',LTERM,';ICLS =',ICLS     CLS04020
          PRINT*,'ISET =',ISET,';LTERM =',LTERM,';ICLS =',ICLS         CLS04030
          DO 500 JCLS=1,NCLS                                            CLS04040
C                                                                       CLS04050
C          THE FOLLOWING 5 STATEMENTS ARE USED FOR INTERNAL CHECKING    CLS04060
C                                                                       CLS04070
C          WRITE(15,*)'JCLS =',JCLS                                     CLS04080
C          DO 460 I=1,NTERM                                             CLS04090
C      460 WRITE(15,*)I,XMCTF(I,JCLS)                                   CLS04100
C          DO 470 I=1,NTERM                                             CLS04110
C      470 WRITE(15,471)I,(VCTF(I,J,JCLS),J=1,NTERM)                    CLS04120
      471 FORMAT(I4,8F9.2)                                              CLS04130
C                                                                       CLS04140
C          RESET THE VARIABLES TO '0.0' FOR FUTURE USE                  CLS04150
C                                                                       CLS04160
          XMCTF(LTERM,JCLS)=VLD                                        CLS04170
          DO 480 I=1,NTERM                                             CLS04180
      480 VCTF(I,LTERM,JCLS)=VLD                                       CLS04190
          DO 490 J=1,NTERM                                             CLS04200
      490 VCTF(LTERM,J,JCLS)=VLD                                       CLS04210
      500 CONTINUE                                                      CLS04220
C                                                                       CLS04230
C          THE FOLLOWING 7 STATEMENTS ARE USED FOR INTERNAL CHECKING    CLS04240
C                                                                       CLS04250
C          DO 550 JCLS=1,NCLS                                           CLS04260
C          WRITE(15,*)'JCLS =',JCLS                                     CLS04270
C          DO 530 I=1,NTERM                                             CLS04280
C      530 WRITE(15,*)I,XMCTF(I,JCLS)                                   CLS04290
C          DO 540 I=1,NTERM                                             CLS04300
C      540 WRITE(15,421)I,(VCTF(I,J,JCLS),J=1,NTERM)                    CLS04310
C      550 CONTINUE                                                     CLS04320
      600 CONTINUE                                                      CLS04330
      650 CONTINUE                                                      CLS04340
          STOP                                                          CLS04350
          END                                                           CLS04360
          SUBROUTINE SR1(IFILE,NP1,NFX,RX)                             CLS04370
```

```
C                                                                    CLS04380
C        THIS SUBROUTINE IS USED TO READ THE INPUT DATA              CLS04390
C                                                                    CLS04400
         REAL RX(NP1)                                                CLS04410
         IKX=MOD(NP1,NFX)                                            CLS04420
         IMX=NFX*(NP1/NFX)+1                                         CLS04430
         ILPX=NP1/NFX                                                CLS04440
         IF(ILPX.EQ.0)ILPR1=1                                        CLS04450
         IF(NP1.LT.NFX)GO TO 20                                      CLS04460
         DO 10 I=1,ILPX                                              CLS04470
      10 READ(IFILE,*)(RX(J),J=1+(I-1)*NFX,I*NFX)                    CLS04480
         IF(IKX.EQ.0)GO TO 30                                        CLS04490
      20 READ(IFILE,*)(RX(J),J=IMX,NP1)                              CLS04500
      30 RETURN                                                      CLS04510
         END                                                         CLS04520




         PROGRAM CANONIC                                             CAN00010
         PARAMETER(NTERM=18,MTERM=NTERM*(NTERM+1)/2,NCLS=3,          CAN00020
        +NWK=NTERM*(NTERM+2))                                        CAN00030
         REAL XMT(NTERM,NCLS),VCVT(MTERM,NCLS),CT(NCLS),             CAN00040
        +VCVIT(MTERM,NCLS),D(NTERM),Z(NTERM,NTERM),WK(NWK),          CAN00050
        +WCS(MTERM),ACS(MTERM),WCS1(MTERM),TEST(NTERM,NTERM),        CAN00060
        +T(NTERM,NTERM),WCSI(MTERM),XMO(NTERM)                       CAN00070
         INTEGER NST(NCLS)                                           CAN00080
         DATA IJOB,IFLAG1,IOPT,NIN,NOUT/2,0,3,0,6/                   CAN00090
C                                                                    CAN00100
C     NTERM  = DIMENSIONALITY IN THE CLASS STATISTICS                CAN00110
C     NCLS   = TOTAL NUMBER OF INFORMATION CLASSES                   CAN00120
C                                                                    CAN00130
C     XMT    = MEAN VECTORS FOR ALL CLASSES                          CAN00140
C     VCVT   = COV. MATRICES FOR ALL CLASSES                         CAN00150
C     CT     = VARIABLE USED TO STORE M.L. THRESHOLD PARAMETER       CAN00160
C     VCVIT  = INVERSE COV. MATRICES FOR ALL CLASSES                 CAN00170
C     D      = EIGENVALUES                                           CAN00180
C     Z      = CANONICAL FEATURES                                    CAN00190
C     WK     = WORKING SPACE FOR IMSL ROUTINES                       CAN00200
C     WCS    = WITHIN CLASS SCATTER MATRIX                           CAN00210
C     ACS    = AMONG CLASS SCATTER MATRIX                            CAN00220
C     WCS1   = TEMPORARY STORAGE FOR WCS                             CAN00230
C     WCSI   = INVERSE MATRIX OF WCS                                 CAN00240
C     XMO    = GLOBAL MEAN VECTOR                                    CAN00250
C     TEST   = INTERNAL CHECKING FOR MATRIX INVERSION ACCURACY       CAN00260
C                                                                    CAN00270
C                                                                    CAN00280
C                                                                    CAN00290
C--->>CHOOSE OR TYPE IN THE CORRECT NUMBERS OF SAMPLES IN THE DATA SETS CAN00300
C                                                                    CAN00310
C ------------------------------------------------------------------CAN00320
C   NSET   F1      NP2     A       B       C     DACO    EXNU    RUSE  CAN00330
C     1  M2611K1   832  WW:141 SF:414 GS:277 760928 76102207   1-1622 CAN00340
C     2  M2611K2  1551  WW:658 SF:211 UC:682 770503 77102207 6515-8096 CAN00350
C     3  M2611K3  1477  WW:677 SF:643 GS:157 770626 77102207 8097-9691 CAN00360
C     4  M2614N1  1265  SW:664 SF:437 NP:164 770508 77102217   1-1396 CAN00370
```

```
C       5  M2614N2   1239   SW:787 SF:291 NP:161 770629 77102217 2777-4141  CAN00380
C       6  M2614N3   1444   SW:931 SF:330 NP:183 770804 77102217 5426-6993  CAN00390
C                                                                           CAN00400
C       DATA NST/141,414,277,658,211,682,677,643,157/                      CAN00410
C       DATA NST/141,414,277,658,211,682,677,643,157,                      CAN00420
C      +664,437,164,787,291,161,931,330,183/                               CAN00430
C       DATA NST/664,437,164,787,291,161,931,330,183/                      CAN00440
C       DATA NST/141,414,277,658,211,682,677/                              CAN00450
C       DATA NST/587,216,121/                                              CAN00460
        DATA NST/658,211,682/                                              CAN00470
C                                                                          CAN00480
C    ------------------------------------------------------------------CAN00490
C       THE FOLLOWING DATA 'NST' ARE USED FOR SOIL ORDER DATA SET. 'SO'    CAN00500
C                                                                          CAN00510
C       NP2=479; MOL ALF EN AR UL IN SP VE H OX UNCLASSIFIED               CAN00520
C       DATA NST/154,113,78,52,45,37,30,11,8,11,32/                        CAN00530
C       DATA NST/154,113,78,52,45,97/                                      CAN00540
C       DATA NST/154,113,78,52,45,37/                                      CAN00550
C                                                                          CAN00560
C    ------------------------------------------------------------------CAN00570
C                                                                          CAN00580
C       THE FOLLOWING DATA 'NST' IS USED FOR SOIL 'OM1' DATA SET           CAN00590
C       I.E. (1) MOLLISOL, OR (2) ALFISOL, AND GROUP SAMPLES               CAN00600
C       ACCORDING TO THEIR ORGANIC MATERIAL: % WEIGHT                      CAN00610
C       CLASS 1 TO 6 :  NP2 = 255                                          CAN00620
C       CLS1 : .11% .GE. OM .LE. 1.5%  :  #  1 -> # 51                     CAN00630
C       CLS2 : 1.5% .GT. OM .LE. 2.0%  :  # 52 -> # 104                    CAN00640
C       CLS3 : 2.0% .GT. OM .LE. 2.5%  :  # 105 -> # 138                   CAN00650
C       CLS4 : 2.5% .GT. OM .LE. 3.5%  :  # 139 -> # 183                   CAN00660
C       CLS5 : 3.5% .GT. OM .LE. 5.0%  :  # 184 -> # 222                   CAN00670
C       CLS6 : 5.0% .GT. OM .LE. 10.12% : # 223 -> # 255                   CAN00680
C                                                                          CAN00690
C       DATA NST/51,54,33,45,39,33/                                        CAN00700
C                                                                          CAN00710
C       DATA 'S2A' :  ANOTHER TEST GROUPED BY THE SAME OM RANGES AS 'OM2'  CAN00720
C       OM PERCENTAGE :  0,1; 1,2; 2,3; 3,4; 4,6; 6 AND ABOVE              CAN00730
C                                                                          CAN00740
C       DATA NST/26,78,64,32,55/                                          CAN00750
C                                                                          CAN00760
C                                                                          CAN00770
C    ------------------------------------------------------------------  CAN00780
C       THE FOLLOWING DATA 'NST' IS USED FOR 'OM2' DATA SET                CAN00790
C       ACCORDING TO THEIR ORGANIC MATERIAL: % WEIGHT                      CAN00800
C       CLASS 1 TO 6 :  NP2 = 514                                          CAN00810
C       CLS1 : .08% .GE. OM .LE. 1.0%  :  #  1 -> # 82                     CAN00820
C       CLS2 : 1.0% .GT. OM .LE. 2.0%  :  # 83 -> # 217                    CAN00830
C       CLS3 : 2.0% .GT. OM .LE. 3.0%  :  # 218 -> # 337                   CAN00840
C       CLS4 : 3.0% .GT. OM .LE. 4.0%  :  # 338 -> # 391                   CAN00850
C       CLS5 : 4.0% .GT. OM .LE. 6.0%  :  # 392 -> # 450                   CAN00860
C       CLS6 : 6.0% .GT. OM .LE. 84.79% : # 451 -> # 514                   CAN00870
C                                                                          CAN00880
C       DATA NST/82,135,120,54,59,64/                                      CAN00890
C                                                                          CAN00900
C       DATA NST/82,135,120,54,123/                                        CAN00910
C       DATA NST/44,31,18,23,24,51,37,27/                                  CAN00920
C       DATA NST/83,57,94,31,37,59,103,26,24/                              CAN00930
C       DATA NST/103,26,24/                                                CAN00940
```

```
C     ------------------------------------------------------------  CAN00950
C       THE FOLLOWING DATA 'NST' IS USED FOR SOIL IRON OXIDE 'IO' DATA SETCAN00960
C       ACCORDING TO THEIR FE2O3 % WEIGHT                           CAN00970
C       CLASS 1 TO 6 :   NP2 = 467                                  CAN00980
C       CLS1 : .02% .GE. FE2O3 .LE. 0.4%  :  #  1 -> # 102          CAN00990
C       CLS2 : 0.4% .GT. FE2O3 .LE. 0.6%  :  # 103 -> # 175         CAN01000
C       CLS3 : 0.6% .GT. FE2O3 .LE. 0.8%  :  # 176 -> # 244         CAN01010
C       CLS4 : 0.8% .GT. FE2O3 .LE. 1.2%  :  # 245 -> # 349         CAN01020
C       CLS5 : 1.2% .GT. FE2O3 .LE. 1.6%  :  # 350 -> # 401         CAN01030
C       CLS6 : 1.6% .GT. FE2O3 .LE. 25.60% : # 402 -> # 467         CAN01040
C                                                                   CAN01050
C       DATA NST/102,73,69,105,52,66/                              CAN01060
C                                                                   CAN01070
C     ------------------------------------------------------------  CAN01080
C       THE FOLLOWING DATA 'NST' IS USED FOR SOIL TEXTURE 'ST' DATA SET CAN01090
C       ACCORDING TO THEIR SAND-SILT-CLAY % CONTENT                 CAN01100
C       CLASS 1 TO 6 :   NP2 = 483; DETAILS : SEE FILE ( S5L.DATA.C1) CAN01110
C                                                                   CAN01120
C       DATA NST/40,63,76,93,68,143/                               CAN01130
C                                                                   CAN01140
C     ------------------------------------------------------------  CAN01150
C       THE FOLLOWING DATA 'NST' IS USED FOR S.D. VEGETATION DATA   CAN01160
C                                                                   CAN01170
C       DATA NST/225,61,292,469, 82,182,63,103, 39,39,217,51,      CAN01180
C      +393,441,80,88, 88,41,32,26, 118,43,121,44, 45,102,66,89,   CAN01190
C      +78,53,147,39, 24,42,119,69, 76,96,107,154, 28,19/          CAN01200
C                                                                   CAN01210
C     ------------------------------------------------------------  CAN01220
C       THE FOLLOWING DATA 'NST' IS USED FOR IOWA VEGETATION DATA   CAN01230
C                                                                   CAN01240
C       DATA NST/514,41, 517,36,32, 621,517,45, 610,485,21,        CAN01250
C      +437,377,22, 190,172,25, 650,568,42, 435,417,44, 393,267/   CAN01260
C------------------------------------------------------------------CAN01270
C                                                                   CAN01280
C       11 = CLASS STATISTICS; 12 = CANONICAL FEATURES             CAN01290
C                                                                   CAN01300
        OPEN(11)                                                    CAN01310
        OPEN(12)                                                    CAN01320
        REWIND 11                                                   CAN01330
        REWIND 12                                                   CAN01340
C                                                                   CAN01350
C       SET THE INPUT&OUTPUT DO LOOP PARAMETERS                     CAN01360
C                                                                   CAN01370
        IK1=MOD(NCLS,6)                                             CAN01380
        IM1=6*(NCLS/6)+1                                            CAN01390
        IK2=MOD(NTERM,5)                                            CAN01400
        IM2=5*(NTERM/5)+1                                           CAN01410
        IK3=MOD(NTERM,16)                                           CAN01420
        IM3=16*(NTERM/16)+1                                         CAN01430
        ILP1=NCLS/6                                                 CAN01440
        IF(ILP1.EQ.0)ILP1=1                                         CAN01450
        ILP2=NTERM/5                                                CAN01460
        IF(ILP2.EQ.0)ILP2=1                                         CAN01470
        ILP3=NTERM/16                                               CAN01480
        IF(ILP3.EQ.0)ILP3=1                                         CAN01490
C                                                                   CAN01500
C       SET THE IMSL INPUT&OUTPUT TO THE FEATURE DESIGNER ( SCREEN ) CAN01510
```

```
C                                                                    CAN01520
      CALL UGETIO(IOPT,NIN,NOUT)                                     CAN01530
      DO 130 LTERM=1,NTERM                                           CAN01540
      KTERM=LTERM*(LTERM+1)/2                                        CAN01550
C                                                                    CAN01560
C     READ IN CLASS STATISTICS                                      CAN01570
C                                                                    CAN01580
      DO 30 ITERM=1,LTERM                                           CAN01590
      IF(NCLS.LT.6)GO TO 20                                         CAN01600
      DO 10 IL=1,ILP1                                               CAN01610
   10 READ(11,*)(XMT(ITERM,JCLS),JCLS=1+(IL-1)*6,IL*6)             CAN01620
      IF(IK1.EQ.0)GO TO 30                                          CAN01630
   20 READ(11,*)(XMT(ITERM,JCLS),JCLS=IM1,NCLS)                     CAN01640
   30 CONTINUE                                                      CAN01650
      DO 60 ITERM=1,KTERM                                           CAN01660
      IF(NCLS.LT.6)GO TO 50                                         CAN01670
      DO 40 IL=1,ILP1                                               CAN01680
   40 READ(11,*)(VCVT(ITERM,JCLS),JCLS=1+(IL-1)*6,IL*6)           CAN01690
      IF(IK1.EQ.0)GO TO 60                                          CAN01700
   50 READ(11,*)(VCVT(ITERM,JCLS),JCLS=IM1,NCLS)                    CAN01710
   60 CONTINUE                                                      CAN01720
      IF(NCLS.LT.6)GO TO 80                                         CAN01730
      DO 70 IL=1,ILP1                                               CAN01740
   70 READ(11,*)(CT(ICLS),ICLS=1+(IL-1)*6,IL*6)                    CAN01750
      IF(IK1.EQ.0)GO TO 90                                          CAN01760
   80 READ(11,*)(CT(ICLS),ICLS=IM1,NCLS)                            CAN01770
   90 DO 120 ITERM=1,KTERM                                          CAN01780
      IF(NCLS.LT.6)GO TO 110                                        CAN01790
      DO 100 IL=1,ILP1                                              CAN01800
  100 READ(11,*)(VCVIT(ITERM,JCLS),JCLS=1+(IL-1)*6,IL*6)          CAN01810
      IF(IK1.EQ.0)GO TO 120                                         CAN01820
  110 READ(11,*)(VCVIT(ITERM,JCLS),JCLS=IM1,NCLS)                   CAN01830
  120 CONTINUE                                                      CAN01840
  130 CONTINUE                                                      CAN01850
C                                                                    CAN01860
C     FIND WITHIN CLASS SCATTER MATRIX                              CAN01870
C                                                                    CAN01880
      CALL FWCS(VCVT,MTERM,NST,NCLS,WCS)                            CAN01890
C     CALL USWSM(' WCS MATRIX IS ',15,WCS,NTERM,1)                 CAN01900
C                                                                    CAN01910
C     FIND AMONG CLASS SCATTER MATRIX                               CAN01920
C                                                                    CAN01930
      CALL FACS(XMT,NTERM,MTERM,NST,NCLS,ACS,XMO)                  CAN01940
C     CALL USWSM(' ACS MATRIX IS ',15,ACS,NTERM,1)                 CAN01950
C                                                                    CAN01960
C     FIND CANONICAL FEATURES                                       CAN01970
C                                                                    CAN01980
      CALL EIGZS(ACS,WCS,NTERM,IJOB,D,Z,NTERM,WK,IER)             CAN01990
C     CALL USWFV('CANONIC EVALUES',15,D,NTERM,1,1)                 CAN02000
C     CALL USWSM('CANONIC EVECTOR',15,Z,NTERM,1)                   CAN02010
C                                                                    CAN02020
C     INTERNAL CHECKING FOR MATRIX INVERSION ACCURACY              CAN02030
C                                                                    CAN02040
      CALL SCOPY(MTERM,WCS,1,WCS1,1)                                CAN02050
      CALL LINV1P(WCS1,NTERM,WCSI,IDGT,D1,D2,IER1)                 CAN02060
      CALL VMULSS(WCSI,ACS,NTERM,TEST,NTERM)                       CAN02070
      CALL FTRACE(TEST,NTERM,TRACE)                                CAN02080
```

```
C                                                                    CAN02090
C       SEND THE ACCURACY COMMENTS TO THE SCREEN                     CAN02100
C                                                                    CAN02110
        IF(IER.NE.0.OR.WK(1).GE.1.0)GO TO 140                        CAN02120
        WRITE(*,3)IER,WK(1)                                          CAN02130
        GO TO 150                                                    CAN02140
  140 WRITE(*,2)IER,WK(1)                                            CAN02150
    1 FORMAT(5E15.7)                                                 CAN02160
    2 FORMAT(' PERFORMANCE OF "EIGZS" IS POOR, IER =',I5,            CAN02170
    +'; WK(1) =',E15.7)                                              CAN02180
    3 FORMAT(' PERFORMANCE OF "EIGZS" IS GOOD, IER =',I5,            CAN02190
    +'; WK(1) =',E15.7)                                              CAN02200
  150 DO 170 I=1,NTERM                                               CAN02210
        IF(D(I).LE.0.0)GO TO 160                                     CAN02220
        GO TO 170                                                    CAN02230
  160 WRITE(*,4)I,D(I)                                               CAN02240
    4 FORMAT(' EIGEN VALUE IS "< = 0.0" AT I =', I5,                 CAN02250
    +' WHERE D(I) =',E15.7)                                          CAN02260
        IFLAG1=IFLAG1+1                                              CAN02270
  170 CONTINUE                                                       CAN02280
        IF(IFLAG1.GT.0)GO TO 180                                     CAN02290
        WRITE(*,6)                                                   CAN02300
        GO TO 190                                                    CAN02310
  180 WRITE(*,5)IFLAG1                                               CAN02320
    5 FORMAT(' THERE ARE',I5,' NEGATIVE OR ZERO EIGEN VALUES ')      CAN02330
    6 FORMAT(' ALL EIGEN VALUES ARE GREATER THAN ZERO ')            CAN02340
  190 CALL VABSMF(D,NTERM,1,SUM)                                     CAN02350
        IF(ABS(TRACE-SUM).GT.1.0E-1)GO TO 200                        CAN02360
        WRITE(*,8)TRACE,SUM                                          CAN02370
        GO TO 210                                                    CAN02380
  200 WRITE(*,7)TRACE,SUM                                            CAN02390
    7 FORMAT(' ACCURACY OF "EIGZS" IS POOR, TRACE =',E15.7,          CAN02400
    +'; SUM =',E15.7)                                                CAN02410
    8 FORMAT(' ACCURACY OF "EIGZS" IS GOOD, TRACE =',E15.7,          CAN02420
    +'; SUM =',E15.7)                                                CAN02430
C                                                                    CAN02440
C       SEND THE FINAL RESULTS TO THE CANONICAL FEATURE FILE         CAN02450
C                                                                    CAN02460
  210 WRITE(12,9)TRACE,SUM                                           CAN02470
    9 FORMAT(2E15.7)                                                 CAN02480
        IF(NTERM.LT.5)GO TO 230                                      CAN02490
        DO 220 I=1,ILP2                                              CAN02500
  220 WRITE(12,1)(D(NTERM+1-J),J=1+(I-1)*5,I*5)                      CAN02510
        IF(IK2.EQ.0)GO TO 240                                        CAN02520
  230 WRITE(12,1)(D(NTERM+1-J),J=IM2,NTERM)                          CAN02530
  240 DO 270 J=1,NTERM                                               CAN02540
        IF(NTERM.LT.5)GO TO 260                                      CAN02550
        DO 250 I=1,ILP2                                              CAN02560
  250 WRITE(12,1)(Z(K,NTERM+1-J),K=1+(I-1)*5,I*5)                    CAN02570
        IF(IK2.EQ.0)GO TO 270                                        CAN02580
  260 WRITE(12,1)(Z(K,NTERM+1-J),K=IM2,NTERM)                        CAN02590
  270 CONTINUE                                                       CAN02600
        CALL VMULSF(WCS,NTERM,Z,NTERM,NTERM,TEST,NTERM)              CAN02610
        NTM=NTERM                                                    CAN02620
        CALL VMULFM(Z,TEST,NTM,NTM,NTM,NTM,NTM,T,NTM,IER)            CAN02630
C                                                                    CAN02640
C       SEND THE ACCURACY COMMENTS TO THE SCREEN                     CAN02650
```

```
C                                                                      CAN02660
      PRINT*,' THE FOLLOWING MATRIX MUST BE AN IDENTITY MATRIX'        CAN02670
      IF(NTERM.LT.16)GO TO 290                                         CAN02680
      DO 280 IL=1,ILP3                                                 CAN02690
      DO 280 I=1,NTERM                                                 CAN02700
  280 WRITE(*,281)(T(I,J),J=1+(IL-1)*16,IL*16)                         CAN02710
  281 FORMAT(16F5.2)                                                   CAN02720
      IF(IK3.EQ.0)GO TO 310                                            CAN02730
  290 DO 300 I=1,NTERM                                                 CAN02740
  300 WRITE(*,281)(T(I,J),J=IM3,NTERM)                                 CAN02750
  310 STOP                                                             CAN02760
      END                                                              CAN02770
      SUBROUTINE FWCS(VCVT,MTERM,NST,NCLS,WCS)                         CAN02780
C                                                                      CAN02790
C     THIS SUBROUTINE FINDS WITHIN CLASS SCATTER MATRIX               CAN02800
C                                                                      CAN02810
      REAL VCVT(MTERM,NCLS),WCS(MTERM)                                 CAN02820
      INTEGER NST(NCLS)                                                CAN02830
      NX1=0                                                            CAN02840
      DO 10 I=1,NCLS                                                   CAN02850
   10 NX1=NX1+NST(I)                                                   CAN02860
      DO 30 I=1,MTERM                                                  CAN02870
      X1=0.0                                                           CAN02880
      DO 20 J=1,NCLS                                                   CAN02890
      X2=FLOAT(NST(J))-1.0                                             CAN02900
   20 X1=X1+X2*VCVT(I,J)/FLOAT(NX1)                                    CAN02910
   30 WCS(I)=X1                                                        CAN02920
      RETURN                                                           CAN02930
      END                                                              CAN02940
      SUBROUTINE FACS(XMT,NTERM,MTERM,NST,NCLS,ACS,XMO)                CAN02950
C                                                                      CAN02960
C     THIS SUBROUTINE FINDS AMONG CLASS SCATTER MATRIX                CAN02970
C                                                                      CAN02980
      REAL XMT(NTERM,NCLS),ACS(MTERM),XMO(NTERM)                       CAN02990
      INTEGER NST(NCLS)                                                CAN03000
      NX1=0                                                            CAN03010
      DO 10 I=1,NCLS                                                   CAN03020
   10 NX1=NX1+NST(I)                                                   CAN03030
      DO 30 I=1,NTERM                                                  CAN03040
      X1=0.0                                                           CAN03050
      DO 20 J=1,NCLS                                                   CAN03060
      X2=FLOAT(NST(J))                                                 CAN03070
   20 X1=X1+X2*XMT(I,J)/FLOAT(NX1)                                     CAN03080
   30 XMO(I)=X1                                                        CAN03090
      DO 50 I=1,NTERM                                                  CAN03100
      DO 50 J=1,I                                                      CAN03110
      IND=(I-1)*I/2+J                                                  CAN03120
      X1=0.0                                                           CAN03130
      DO 40 ICLS=1,NCLS                                                CAN03140
      X2=FLOAT(NST(ICLS))/FLOAT(NX1)                                   CAN03150
   40 X1=X1+X2*(XMT(I,ICLS)-XMO(I))*(XMT(J,ICLS)-XMO(J))               CAN03160
   50 ACS(IND)=X1                                                      CAN03170
      RETURN                                                           CAN03180
      END                                                              CAN03190
      SUBROUTINE FTRACE(TEST,NTERM,TRACE)                              CAN03200
      REAL TEST(NTERM,NTERM)                                           CAN03210
      TRACE=0.0                                                        CAN03220
```

```
       DO 10 I=1,NTERM                                             CAN03230
    10 TRACE=TRACE+TEST(I,I)                                       CAN03240
       RETURN                                                      CAN03250
       END                                                         CAN03260




       PROGRAM PCFIND                                              PCF00010
       PARAMETER(NTSET=4,NTERM=16,MTERM=NTERM*(NTERM+1)/2,NCLS=3,  PCF00020
      +NSET=1,MSET=1,NDSET=1,NSMAX=100,NZ1=NCLS*NCLS*NTERM,        PCF00030
      +IRES=0,IFIND=1,ICKMV=0,NDTRM=1,NZ2=NCLS*NTERM,NTERMC=15)    PCF00040
C                                                                  PCF00050
C      IFIND = 1 ---> NDTRM CONTROL : LTERM=1,NTERM,NDTRM          PCF00060
C      IFIND = 0 ---> NDTRM DISABLE : LTERM = NTERM ONLY           PCF00070
C-->>  IRES  = 1 ---> NSMAX MUST EXCEED MAX(NST(I))   <<----- NOTES!!!  PCF00080
C      IRES  = 0 ---> NSMAX CONTROL : SUBROUTINE GGNSM             PCF00090
C                                                                  PCF00100
C      NTERMC > OR = NTERM , WHERE NTERMC IS USED TO READ ENTIRE   PCF00110
C                 TRANSFORMED DATA; WHILE NTERM IS USED TO DECIDE  PCF00120
C                 HOW MANY OF THEM WILL BE CONTRIBUTED TO PC       PCF00130
C                                                                  PCF00140
C      NTERM = TOTAL NUMBER OF FEATURES USED                      PCF00150
C      NCLS  = TOTAL NUMBER OF INFORMATION CLASSES                PCF00160
C      NSMAX = PRESET MAX. NO. OF SAMPLES FOR ONE CLASS           PCF00170
C                                                                  PCF00180
       REAL XMT(NTERM,NCLS),VCVT(MTERM,NCLS),CT(NCLS),             PCF00190
      +VCVIT(MTERM,NCLS),TVEC(NSMAX,NTERM,NCLS),                   PCF00200
      +VCVIF(NTERM,NTERM),VCV(MTERM),VCVI(MTERM),XM(NTERM),        PCF00210
      +PC(NTERM),QP(NCLS,NTERM),PR(NCLS,NTERM),PX(NCLS),           PCF00220
      +VEC(NSMAX,NTERM),WK(NTERM),X(NTERM),T1(NTERM)               PCF00230
       REAL XMCK(NTERM),VCVCK(MTERM),TX(NTERM),Y(NTERM)            PCF00240
       REAL RVEC(NSMAX,NTERMC,NCLS),AP(NCLS)                       PCF00250
       INTEGER NBR(6),NPC(NCLS,NCLS,NTERM),NST(NCLS)               PCF00260
       CHARACTER*2 XC1                                             PCF00270
       DOUBLE PRECISION DSEED                                      PCF00280
       DATA XC1/'  '/                                              PCF00290
       DATA PC/NTERM*0.0/                                          PCF00300
       DATA QP,PR,PX/NZ2*0.0,NZ2*0.0,NCLS*0.0/                     PCF00310
       DATA DSEED,NPC/5.D0,NZ1*0/                                  PCF00320
       DATA (NBR(I),I=4,6),IOPT,NIN,NOUT/1,0,0,3,0,6/              PCF00330
C                                                                  PCF00340
C      XMT   = MEAN VECTORS FOR ALL CLASSES                       PCF00350
C      VCVT  = COV. MATRICES FOR ALL CLASSES                     PCF00360
C      CT    = M.L. DECISION RULE PARAMETER                      PCF00370
C      VCVIT = INVERSE COV. MATRICES FOR ALL CLASSES             PCF00380
C      TVEC  = GENERATED SAMPLE VECTORS                          PCF00390
C      VCVIF = INVERSE COV. MATRIX IN FULL STORAGE MODE          PCF00400
C      VCV   = COV. MATRIX                                       PCF00410
C      VCVI  = INVERSE COV. MATRIX IN SYMMETRIC STORAGE MODE     PCF00420
C      XM    = MEAN VECTOR                                       PCF00430
C      PC    = PROBABILITY OF CORRECT CLASSIFICATION             PCF00440
C      XMCK  = CHECKING VECTOR FOR MEAN                          PCF00450
C      VCVCK = CHECKING MATRIX FOR COVARIANCES                   PCF00460
C      NBR   = IMSL ROUTINE-USED PARAMETER VECTOR                PCF00470
C      NPC   = CLASSIFICATION RESULT MATRIX                      PCF00480
```

```
C     NST     = STORE THE TOTAL NO. OF SAMPLES FOR EACH CLASS        PCF00490
C                                                                    PCF00500
C     ---------------------------------------------------------------PCF00510
C                                                                    PCF00520
C--->>CHOOSE OR TYPE IN THE CORRECT NUMBERS OF SAMPLES IN THE DATA SETS PCF00530
C                                                                    PCF00540
C     ---------------------------------------------------------------PCF00550
C     NSET    F1     NP2    A       B       C      DACO     EXNU      RUSE  PCF00560
C       1   M2611K1   832  WW:141  SF:414  GS:277  760928  76102207   1-1622 PCF00570
C       2   M2611K2  1551  WW:658  SF:211  UC:682  770503  77102207 6515-8096 PCF00580
C       3   M2611K3  1477  WW:677  SF:643  GS:157  770626  77102207 8097-9691 PCF00590
C       4   M2614N1  1265  SW:664  SF:437  NP:164  770508  77102217   1-1396 PCF00600
C       5   M2614N2  1239  SW:787  SF:291  NP:161  770629  77102217 2777-4141 PCF00610
C       6   M2614N3  1444  SW:931  SF:330  NP:183  770804  77102217 5426-6993 PCF00620
C                                                                    PCF00630
C     DATA NST/141,414,277,658,211,682,677,643,157/                 PCF00640
C     DATA NST/141,414,277,658,211,682,677,643,157,                 PCF00650
C    +664,437,164,787,291,161,931,330,183/                          PCF00660
C     DATA NST/664,437,164,787,291,161,931,330,183/                 PCF00670
C     DATA NST/141,414,277,658,211,682,677/                         PCF00680
C     DATA NST/587,216,121/                                         PCF00690
C     DATA NST/658,211,682/                                         PCF00700
C                                                                    PCF00710
C     ---------------------------------------------------------------PCF00720
C     THE FOLLOWING DATA 'NST' ARE USED FOR SOIL ORDER DATA SET. 'SO' PCF00730
C                                                                    PCF00740
C     NP2=479; MOL ALF EN AR UL IN SP VE H OX UNCLASSIFIED          PCF00750
C     DATA NST/154,113,78,52,45,37,30,11,8,11,32/                   PCF00760
C     DATA NST/154,113,78,52,45,97/                                 PCF00770
C     DATA NST/154,113,78,52,45,37/                                 PCF00780
C                                                                    PCF00790
C     ---------------------------------------------------------------PCF00800
C                                                                    PCF00810
C     THE FOLLOWING DATA 'NST' IS USED FOR SOIL 'OM1' DATA SET      PCF00820
C     I.E. (1) MOLLISOL, OR (2) ALFISOL, AND GROUP SAMPLES          PCF00830
C     ACCORDING TO THEIR ORGANIC MATERIAL: % WEIGHT                 PCF00840
C     CLASS 1 TO 6 :  NP2 = 255                                     PCF00850
C     CLS1 : .11% .GE. OM .LE. 1.5%  :  #  1 -> # 51                PCF00860
C     CLS2 : 1.5% .GT. OM .LE. 2.0%  :  # 52 -> # 104               PCF00870
C     CLS3 : 2.0% .GT. OM .LE. 2.5%  :  # 105 -> # 138              PCF00880
C     CLS4 : 2.5% .GT. OM .LE. 3.5%  :  # 139 -> # 183              PCF00890
C     CLS5 : 3.5% .GT. OM .LE. 5.0%  :  # 184 -> # 222              PCF00900
C     CLS6 : 5.0% .GT. OM .LE. 10.12% : # 223 -> # 255              PCF00910
C                                                                    PCF00920
C     DATA NST/51,54,33,45,39,33/                                   PCF00930
C                                                                    PCF00940
C     DATA 'S2A' :  ANOTHER TEST GROUPED BY THE SAME OM RANGES AS 'OM2' PCF00950
C     OM PERCENTAGE :  0,1; 1,2; 2,3; 3,4; 4,6; 6 AND ABOVE         PCF00960
C                                                                    PCF00970
C     DATA NST/26,78,64,32,55/                                      PCF00980
C                                                                    PCF00990
C                                                                    PCF01000
C     --------------------------------------------------------------- PCF01010
C     THE FOLLOWING DATA 'NST' IS USED FOR 'OM2' DATA SET           PCF01020
C     ACCORDING TO THEIR ORGANIC MATERIAL: % WEIGHT                 PCF01030
C     CLASS 1 TO 6 :  NP2 = 514                                     PCF01040
C     CLS1 : .08% .GE. OM .LE. 1.0%  :  #  1 -> # 82                PCF01050
```

```
C      CLS2 : 1.0% .GT. OM .LE.  2.0%  :  # 83 -> # 217          PCF01060
C      CLS3 : 2.0% .GT. OM .LE.  3.0%  :  # 218 -> # 337         PCF01070
C      CLS4 : 3.0% .GT. OM .LE.  4.0%  :  # 338 -> # 391         PCF01080
C      CLS5 : 4.0% .GT. OM .LE.  6.0%  :  # 392 -> # 450         PCF01090
C      CLS6 : 6.0% .GT. OM .LE. 84.79% :  # 451 -> # 514         PCF01100
C                                                                PCF01110
C      DATA NST/82,135,120,54,59,64/                            PCF01120
C                                                                PCF01130
C      DATA NST/82,135,120,54,123/                              PCF01140
C      DATA NST/44,31,18,23,24,51,37,27/                        PCF01150
C      DATA NST/83,57,94,31,37,59,103,26,24/                    PCF01160
C      DATA NST/103,26,24/                                      PCF01170
C      --------------------------------------------------------- PCF01180
C      THE FOLLOWING DATA 'NST' IS USED FOR SOIL IRON OXIDE 'IO' DATA SETPCF01190
C      ACCORDING TO THEIR FE2O3 % WEIGHT                        PCF01200
C      CLASS 1 TO 6 :  NP2 = 467                                PCF01210
C      CLS1 : .02% .GE. FE2O3 .LE.  0.4%  :  #  1 -> # 102      PCF01220
C      CLS2 : 0.4% .GT. FE2O3 .LE.  0.6%  :  # 103 -> # 175     PCF01230
C      CLS3 : 0.6% .GT. FE2O3 .LE.  0.8%  :  # 176 -> # 244     PCF01240
C      CLS4 : 0.8% .GT. FE2O3 .LE.  1.2%  :  # 245 -> # 349     PCF01250
C      CLS5 : 1.2% .GT. FE2O3 .LE.  1.6%  :  # 350 -> # 401     PCF01260
C      CLS6 : 1.6% .GT. FE2O3 .LE. 25.60% :  # 402 -> # 467     PCF01270
C                                                                PCF01280
C      DATA NST/102,73,69,105,52,66/                            PCF01290
C                                                                PCF01300
C      --------------------------------------------------------- PCF01310
C      THE FOLLOWING DATA 'NST' IS USED FOR SOIL TEXTURE 'ST' DATA SET PCF01320
C      ACCORDING TO THEIR SAND-SILT-CLAY % CONTENT              PCF01330
C      CLASS 1 TO 6 :  NP2 = 483; DETAILS : SEE FILE ( S5L.DATA.C1) PCF01340
C                                                                PCF01350
C      DATA NST/40,63,76,93,68,143/                             PCF01360
C                                                                PCF01370
C      --------------------------------------------------------- PCF01380
C      THE FOLLOWING DATA 'NST' IS USED FOR S.D. VEGETATION DATA PCF01390
C                                                                PCF01400
C      DATA NST/225,61,292,469, 82,182,63,103,  39,39,217,51,   PCF01410
C     +393,441,80,88,  88,41,32,26, 118,43,121,44,  45,102,66,89, PCF01420
C     +78,53,147,39,  24,42,119,69,  76,96,107,154,  28,19/     PCF01430
C                                                                PCF01440
C      --------------------------------------------------------- PCF01450
C      THE FOLLOWING DATA 'NST' IS USED FOR IOWA VEGETATION DATA PCF01460
C                                                                PCF01470
C      DATA NST/514,41,  517,36,32,  621,517,45,  610,485,21,   PCF01480
C     +437,377,22,  190,172,25,  650,568,42,  435,417,44,  393,267/ PCF01490
C                                                                PCF01500
C      --------------------------------------------------------- PCF01510
C                                                                PCF01520
C      11 = TRANSFORMED DATA; 12 = CLASS STATISCTICS; 13 = PC   PCF01530
C                                                                PCF01540
       OPEN(11)                                                 PCF01550
       OPEN(12)                                                 PCF01560
       OPEN(13)                                                 PCF01570
       REWIND 11                                                PCF01580
       REWIND 12                                                PCF01590
       REWIND 13                                                PCF01600
       NX2=0                                                    PCF01610
       DO 1 I=1,NCLS                                            PCF01620
```

```
    1 NX2=NX2+NST(I)                                              PCF01630
      IF(IRES.EQ.0)GO TO 3                                        PCF01640
      DO 2 I=1,NCLS                                               PCF01650
      NX1=NST(I)                                                  PCF01660
    2 AP(I)=FLOAT(NX1)/FLOAT(NX2)                                 PCF01670
      GO TO 5                                                     PCF01680
    3 DO 4 I=1,NCLS                                               PCF01690
    4 AP(I)=1.0/FLOAT(NCLS)                                       PCF01700
    5 IK=MOD(NCLS,6)                                              PCF01710
C                                                                 PCF01720
C     SET THE INPUT&OUTPUT DO LOOP PARAMETERS                     PCF01730
C                                                                 PCF01740
      IM=6*(NCLS/6)+1                                             PCF01750
      IK1=MOD(NCLS,3)                                             PCF01760
      IM1=3*(NCLS/3)+1                                            PCF01770
      IK2=MOD(NCLS,15)                                            PCF01780
      IM2=15*(NCLS/15)+1                                          PCF01790
      ILP1=NCLS/6                                                 PCF01800
      IF(ILP1.EQ.0)ILP1=1                                        PCF01810
      ILP2=NCLS/3                                                 PCF01820
      IF(ILP2.EQ.0)ILP2=1                                        PCF01830
      ILP3=NCLS/15                                                PCF01840
      IF(ILP3.EQ.0)ILP3=1                                        PCF01850
      IF(IRES.EQ.0)NSAMP=NSMAX                                    PCF01860
      DO 550 ISET=NSET,MSET,NDSET                                 PCF01870
      IF(IRES.EQ.1)CALL RDATA(ISET,RVEC,NSMAX,NTERMC,NCLS,NST)    PCF01880
C                                                                 PCF01890
C     READ IN CLASS STATISTICS                                   PCF01900
C                                                                 PCF01910
      DO 500 LTERM=1,NTERM                                        PCF01920
      KTERM=LTERM*(LTERM+1)/2                                     PCF01930
      DO 30 ITERM=1,LTERM                                         PCF01940
      IF(NCLS.LT.6)GO TO 20                                       PCF01950
      DO 10 IL=1,ILP1                                             PCF01960
   10 READ(12,*)(XMT(ITERM,JCLS),JCLS=1+(IL-1)*6,IL*6)            PCF01970
      IF(IK.EQ.0)GO TO 30                                         PCF01980
   20 READ(12,*)(XMT(ITERM,JCLS),JCLS=IM,NCLS)                    PCF01990
   30 CONTINUE                                                    PCF02000
      DO 60 ITERM=1,KTERM                                         PCF02010
      IF(NCLS.LT.6)GO TO 50                                       PCF02020
      DO 40 IL=1,ILP1                                             PCF02030
   40 READ(12,*)(VCVT(ITERM,JCLS),JCLS=1+(IL-1)*6,IL*6)           PCF02040
      IF(IK.EQ.0)GO TO 60                                         PCF02050
   50 READ(12,*)(VCVT(ITERM,JCLS),JCLS=IM,NCLS)                   PCF02060
   60 CONTINUE                                                    PCF02070
      IF(NCLS.LT.6)GO TO 80                                       PCF02080
      DO 70 IL=1,ILP1                                             PCF02090
   70 READ(12,*)(CT(ICLS),ICLS=1+(IL-1)*6,IL*6)                   PCF02100
      IF(IK.EQ.0)GO TO 90                                         PCF02110
   80 READ(12,*)(CT(ICLS),ICLS=IM,NCLS)                           PCF02120
   90 DO 120 ITERM=1,KTERM                                        PCF02130
      IF(NCLS.LT.6)GO TO 110                                      PCF02140
      DO 100 IL=1,ILP1                                            PCF02150
  100 READ(12,*)(VCVIT(ITERM,JCLS),JCLS=1+(IL-1)*6,IL*6)          PCF02160
      IF(IK.EQ.0)GO TO 120                                        PCF02170
  110 READ(12,*)(VCVIT(ITERM,JCLS),JCLS=IM,NCLS)                  PCF02180
  120 CONTINUE                                                    PCF02190
```

```
        IF(IFIND.EQ.1)GO TO 125                                         PCF02200
        IF(LTERM.NE.NTERM)GO TO 500                                     PCF02210
        IF(IFIND.EQ.0)GO TO 128                                         PCF02220
C                                                                       PCF02230
C       FIND THE PC RESULTS FOR EVERY DTERM INCREMENT                   PCF02240
C                                                                       PCF02250
   125 NX1=LTERM+(NDTRM-1)                                              PCF02260
        NX2=MOD(NX1,NDTRM)                                              PCF02270
C       PRINT*,NX1,NX2                                                  PCF02280
        IF(NX2.NE.0)GO TO 500                                           PCF02290
   128 DO 170 JCLS=1,NCLS                                               PCF02300
        DO 130 I=1,KTERM                                                PCF02310
   130 VCV(I)=VCVT(I,JCLS)                                              PCF02320
        CALL UGETIO(IOPT,NIN,NOUT)                                      PCF02330
C       CALL USWSM(' THE MATRIX IS ',15,VCV,LTERM,1)                    PCF02340
C       NOTE : WK(1) MUST BE 0.0 EVERY TIME TO INITIALIZE ' GGNSM '     PCF02350
C       NOTE : VCV WILL BE CHANGED AFTER ' GGNSM'                       PCF02360
        IF(IRES.EQ.1)NSAMP=NST(JCLS)                                    PCF02370
        IF(IRES.EQ.1)GO TO 145                                          PCF02380
        DO 140 I=1,NTERM                                                PCF02390
   140 WK(I)=0.0                                                        PCF02400
        DSEED=5.D0                                                      PCF02410
C                                                                       PCF02420
C       GENERATE GAUSSIAN SAMPLES ACCORDING TO THE CLASS STATISTICS     PCF02430
C                                                                       PCF02440
        CALL GGNSM(DSEED,NSAMP,LTERM,VCV,NSMAX,VEC,WK,IER)              PCF02450
   145 DO 155 I=1,NSAMP                                                 PCF02460
        DO 155 J=1,LTERM                                                PCF02470
        IF(IRES.EQ.1)GO TO 150                                          PCF02480
        VEC(I,J)=VEC(I,J)+XMT(J,JCLS)                                   PCF02490
C                                                                       PCF02500
C       STORE THE SAMPLES INTO ARRAY 'TVEC'                             PCF02510
C                                                                       PCF02520
        TVEC(I,J,JCLS)=VEC(I,J)                                         PCF02530
        GO TO 155                                                       PCF02540
   150 TVEC(I,J,JCLS)=RVEC(I,J,JCLS)                                    PCF02550
        VEC(I,J)=RVEC(I,J,JCLS)                                         PCF02560
C       PRINT*,JCLS,I,J,TVEC(I,J,JCLS)                                  PCF02570
   155 CONTINUE                                                         PCF02580
        IF(ICKMV.EQ.0)GO TO 170                                         PCF02590
C                                                                       PCF02600
C       CHECK THE MEAN VECTOR AND COV. MATRIX OF THE GENERATED SAMPLES  PCF02610
C       THE MATRIX 'VEC' WILL BE CHANGED AFTER ' BECOVM '              PCF02620
C                                                                       PCF02630
        DO 160 I=1,NTERM                                                PCF02640
   160 TX(I)=0.0                                                        PCF02650
        NBR(1)=LTERM                                                    PCF02660
        NBR(2)=NSAMP                                                    PCF02670
        NBR(3)=NSAMP                                                    PCF02680
C       IF(LTERM.GT.1)GO TO 600                                         PCF02690
        CALL BECOVM(VEC,NSMAX,NBR,TX,XMCK,VCVCK,IER)                    PCF02700
C                                                                       PCF02710
C       SEND THE CHECKING RESULTS TO THE SCREEN IF NEEDED               PCF02720
C                                                                       PCF02730
C       CALL USWFV(' THE VECTOR IS ',15,XMCK,LTERM,1,1)                 PCF02740
C       CALL USWSM(' THE MATRIX IS ',15,VCVCK,LTERM,1)                  PCF02750
   170 CONTINUE                                                         PCF02760
```

```
C                                                                       PCF02770
C       START CLASSIFICATION JOB FOR EACH CLASS SAMPLES                 PCF02780
C                                                                       PCF02790
        DO 230 JCLS=1,NCLS                                              PCF02800
        IF(IRES.EQ.1)NSAMP=NST(JCLS)                                    PCF02810
        PRINT*,LTERM,JCLS,NSAMP                                         PCF02820
        DO 230 ISAMP=1,NSAMP                                            PCF02830
        DO 180 J=1,LTERM                                                PCF02840
    180 Y(J)=TVEC(ISAMP,J,JCLS)                                         PCF02850
        DO 220 KCLS=1,NCLS                                              PCF02860
C       THE FOLLOWING IS NEEDED SINCE X HAS BEEN CHANGED FOR EVERY KCLS! PCF02870
        DO 190 I=1,LTERM                                                PCF02880
    190 X(I)=Y(I)                                                       PCF02890
        DO 200 I=1,KTERM                                                PCF02900
    200 VCVI(I)=VCVIT(I,KCLS)                                           PCF02910
        CALL VCVTSF(VCVI,LTERM,VCVIF,NTERM)                             PCF02920
        DO 210 I=1,LTERM                                                PCF02930
    210 XM(I)=XMT(I,KCLS)                                               PCF02940
        CALL SAXPY(LTERM,-1.,XM,1,X,1)                                  PCF02950
        CALL VMULFM(X,VCVIF,LTERM,1,LTERM,NTERM,NTERM,T1,1,IER)         PCF02960
        CALL VMULFF(T1,X,1,LTERM,1,1,NTERM,T2,1,IER)                    PCF02970
        T3=EXP(-0.5*T2)                                                 PCF02980
    220 PX(KCLS)=AP(KCLS)*CT(KCLS)*T3                                   PCF02990
C                                                                       PCF03000
C       PERFORM M.L. DECISION RULE                                      PCF03010
C                                                                       PCF03020
        CALL VABMXF(PX(1),NCLS,1,IMAX,BIG)                              PCF03030
        NPC(JCLS,IMAX,LTERM)=NPC(JCLS,IMAX,LTERM)+1                     PCF03040
C       CALL VABSMF(PX,NCLS,1,DEN)                                      PCF03050
C       Q=BIG/DEN                                                       PCF03060
C       WRITE(13,*)JCLS,ISAMP,IMAX,NPC(JCLS,IMAX,LTERM)                 PCF03070
C       WRITE(13,*)(PX(I),I=1,NCLS),IMAX,BIG                            PCF03080
    230 CONTINUE                                                        PCF03090
C                                                                       PCF03100
C       FIND PROBABILITY OF CORRECT CLASSIFICATION PC FROM NPC          PCF03110
C                                                                       PCF03120
        NC1=0                                                           PCF03130
        NC2=0                                                           PCF03140
        DO 240 I=1,NCLS                                                 PCF03150
        IF(IRES.EQ.0)NST(I)=NSMAX                                       PCF03160
        PR(I,LTERM)=(FLOAT(NPC(I,I,LTERM)))/FLOAT(NST(I))               PCF03170
        NC1=NC1+NPC(I,I,LTERM)                                          PCF03180
    240 NC2=NC2+NST(I)                                                  PCF03190
        IF(IRES.EQ.0)NC2=NSMAX*NCLS                                     PCF03200
        PC(LTERM)=(FLOAT(NC1))/FLOAT(NC2)                               PCF03210
        IF(NCLS.LT.3)GO TO 260                                          PCF03220
C                                                                       PCF03230
C       SEND THE RESULTS TO THE SCREEN                                  PCF03240
C                                                                       PCF03250
        DO 250 IL=1,ILP2                                                PCF03260
    250 WRITE(*,*)ISET,LTERM,(PR(I,LTERM),I=1+(IL-1)*3,IL*3)            PCF03270
        IF(IK1.EQ.0)GO TO 270                                           PCF03280
    260 WRITE(*,*)ISET,LTERM,(PR(I,LTERM),I=IM1,NCLS)                   PCF03290
    270 PRINT*,ISET,LTERM,PC(LTERM)                                     PCF03300
C                                                                       PCF03310
C       SEND THE RESULTS TO THE PC FILE                                 PCF03320
C                                                                       PCF03330
```

```
      WRITE(13,*)' LTERM = ',LTERM                             PCF03340
      IF(NCLS.LT.6)GO TO 290                                   PCF03350
      DO 280 IL=1,ILP1                                         PCF03360
 280  WRITE(13,301)(PR(I,LTERM),I=1+(IL-1)*6,IL*6)             PCF03370
      IF(IK.EQ.0)GO TO 300                                     PCF03380
 290  WRITE(13,301)(PR(I,LTERM),I=IM,NCLS)                     PCF03390
 300  WRITE(13,301)PC(LTERM)                                   PCF03400
 301  FORMAT(6F13.5)                                           PCF03410
C                                                              PCF03420
C---< RESET ALL RELATED VARIABLES  >-----                     PCF03430
C     THE FOLLOWING ZEROING PROCEDURES ARE 'ABSOLUTELY' NEEDED!! PCF03440
C     THIS IS DONE FOR EVERY " LTERM = 1, NTERM "             PCF03450
C                                                              PCF03460
      DO 310 K=1,NCLS                                          PCF03470
      DO 310 I=1,NSMAX                                         PCF03480
      DO 310 J=1,NTERM                                         PCF03490
 310  TVEC(I,J,K)=0.0                                          PCF03500
      DO 320 I=1,NCLS                                          PCF03510
      DO 320 J=1,NTERM                                         PCF03520
      QP(I,J)=0.0                                              PCF03530
 320  PR(I,J)=0.0                                              PCF03540
      DO 330 I=1,NTERM                                         PCF03550
 330  PC(I)=0.0                                                PCF03560
      IF(NCLS.LT.15) GO TO 360                                 PCF03570
C                                                              PCF03580
C     SEND THE FINAL CLASSIFICATION MATRIX NPC TO THE PC FILE PCF03590
C                                                              PCF03600
      DO 350 J=1,ILP3                                          PCF03610
      DO 340 I=1,NCLS                                          PCF03620
 340  WRITE(13,341)I,(NPC(I,K,LTERM),K=1+(J-1)*15,J*15)        PCF03630
 341  FORMAT(I3,2X,15I5)                                       PCF03640
      WRITE(13,342)XC1                                         PCF03650
 342  FORMAT(A2)                                               PCF03660
 350  CONTINUE                                                 PCF03670
      IF(IK2.EQ.0)GO TO 500                                    PCF03680
 360  DO 370 I=1,NCLS                                          PCF03690
 370  WRITE(13,341)I,(NPC(I,K,LTERM),K=IM2,NCLS)               PCF03700
 500  CONTINUE                                                 PCF03710
      DO 510 I=1,NCLS                                          PCF03720
      DO 510 J=1,NCLS                                          PCF03730
      DO 510 K=1,NTERM                                         PCF03740
 510  NPC(I,J,K)=0                                             PCF03750
 550  CONTINUE                                                 PCF03760
C                                                              PCF03770
C     THE FOLLOWING STATEMENT IS USED FOR INTERNAL CHECKING   PCF03780
C                                                              PCF03790
C 600 STOP                                                     PCF03800
      STOP                                                     PCF03810
      END                                                      PCF03820
      SUBROUTINE RDATA(LSET,RVEC,NSMAX,NTERMC,NCLS,NST)        PCF03830
      REAL RVEC(NSMAX,NTERMC,NCLS)                             PCF03840
      INTEGER NST(NCLS)                                        PCF03850
      IKX=MOD(NTERMC,5)                                        PCF03860
      IMX=5*(NTERMC/5)+1                                       PCF03870
      ILPX=NTERMC/5                                            PCF03880
      IF(ILPX.EQ.0)ILPX=1                                      PCF03890
      IFILE1=11+(LSET-1)*10                                    PCF03900
```

```
      DO 40 K=1,NCLS                                           PCF03910
      N1=NST(K)                                                PCF03920
      PRINT*,'KCLS = ',K,'; NSAMP = ',N1                       PCF03930
      DO 30 I=1,N1                                             PCF03940
      IF(NTERMC.LT.5)GO TO 20                                  PCF03950
      DO 10 J1=1,ILPX                                          PCF03960
   10 READ(IFILE1,*)(RVEC(I,J,K),J=1+(J1-1)*5,J1*5)            PCF03970
      IF(IKX.EQ.0)GO TO 30                                     PCF03980
   20 READ(IFILE1,*)(RVEC(I,J,K),J=IMX,NTERMC)                 PCF03990
   30 CONTINUE                                                 PCF04000
   40 CONTINUE                                                 PCF04010
      RETURN                                                   PCF04020
      END                                                      PCF04030
```