A Probabilistic Classification Algorithm With Soft Classification Output

Rhonda Phillips

Dissertation submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor Of Philosophy
in
Computer Science

March 30, 2009
Blacksburg, Virginia

Keywords: parallel processing, remote sensing, high performance computing, data reduction, classification, cluster evaluation

A PROBABILISTIC CLASSIFICATION ALGORITHM WITH SOFT CLASSIFICATION OUTPUT

by

Rhonda Phillips

(ABSTRACT)

This thesis presents a shared memory parallel version of the hybrid classification algorithm IGSCR (iterative guided spectral class rejection), a novel data reduction technique that can be used in conjunction with PIGSCR (parallel IGSCR), a noise removal method based on the maximum noise fraction (MNF), and a continuous version of IGSCR (CIGSCR) that outputs soft classifications. All of the above are either classification algorithms or preprocessing algorithms necessary prior to the classification of high dimensional, noisy images. PIGSCR was developed to produce fast and portable code using Fortran 95, OpenMP, and the Hierarchical Data Format version 5 (HDF5) and accompanying data access library. The feature reduction method introduced in this thesis is based on the singular value decomposition (SVD). This feature reduction technique demonstrated that SVD-based feature reduction can lead to more accurate IGSCR classifications than PCA-based feature reduction. This thesis describes a new algorithm used to adaptively filter a remote sensing dataset based on signal-to-noise ratios (SNRs) once the maximum noise fraction (MNF) has been applied. The adaptive filtering scheme improves image quality as shown by estimated SNRs and classification accuracy improvements greater than 10%. The continuous iterative guided spectral class rejection (CIGSCR) classification method is based on the iterative guided spectral class rejection (IGSCR) classification method for remotely sensed data. Both CIGSCR and IGSCR use semisupervised clustering to locate clusters that are associated with classes in a classification scheme. This type of semisupervised classification method is particularly useful in remote sensing where datasets are large, training data are difficult to acquire, and clustering makes the identification of subclasses adequate for training purposes less difficult. Experimental results indicate that the soft classification output by CIGSCR is reasonably accurate (when compared to IGSCR), and the fundamental algorithmic changes in CIGSCR (from IGSCR) result in CIGSCR being less sensitive to input parameters that influence iterations.

ACKNOWLEDGEMENTS

First and foremost, I thank my advisor, Dr. Layne Watson, for taking a chance on me even though I had no research experience, for showing me how to do research, teaching me how to write, and guiding me through the Ph.D. process. Similarly, I would like to Dr. Randolph Wynne for being such a great mentor and committee member, and for calling me "soon to be Dr. Phillips" four years before I actually became Dr. Phillips and "very soon to be Dr. Phillips" for the past two years. I would like to thank Drs. Yang Cao, Calvin Ribbens, Adrian Sandu, and Elisa Sotelino for serving on my thesis committee and giving me advice. I would like to thank Dr. Christine Blinn for her assistance, GIS/remote sensing expertise, and friendship. Furthermore, I would like to thank Dr. Naren Ramakrishnan for his machine learning expertise and advice, and Dr. Valerie Thomas for her assistance in the processing of hyperspectral imagery. I would like to thank Virginia Tech Advanced Research Computing for the use of their shared memory computers that were used to obtain results listed in this thesis. I would like to recognize the Virginia Tech College of Engineering and Computer Science Department for providing financial support through the College of Engineering Teaching Fellowship that enabled me to complete my degree.

During the long course of my graduate studies, several of my peers in the department have provided moral support and camaraderie. Thanks to Patrick for the coffee and Sunkist, Andy for the excellent Chinese food, Ted, Jingwei, Zhenwei, and Manjula for being great labmates, Jeremy for good conversations about graduate school and academia, Pardha for the racquetball, Hari, Tejinder, and Manas for great Indian food, Dan for losing so many bets and owing me lunches, and Uma, Andrew, Patrick, Amrita, and Scott for making membership in Upsilon Pi Epsilon so enjoyable. Thanks also to Dr. Balci for being so involved in UPE and providing recommendation letters.

I would like to acknowledge my family, who contributed to my earning a Ph.D. My father taught me the value of hard work, which was certainly necessary in graduate school. My mother read to me every night, and taught me the value of reading and imagination. Without those things, I would never had considered graduate school and higher education. My sister, Cindi, and my brother, Keith, humored me by being my first students, helping me to discover that I loved teaching. My sister gave me experience with an enthusiastic learner, and my brother gave me valuable experience with a challenging and reluctant learner.

Finally, I would like to acknowledge my husband, Matthew, who has supported me every day of graduate school. He always seemed to believe in my ability, even when I started to question it. He has provided moral support through challenges, and he helped me celebrate when I finally finished.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1: INTRODUCTION

Classification is an essential tool in the analysis of remotely sensed images. Classification provides a foundation for determining the annual rate of deforestation of the Amazon rainforest, determining the change in polar ice, examining the reappearance of forest in former mining sites, and many other natural resource management applications. Accurate classifications require extensive preprocessing of images to rectify and correct for atmospheric conditions. Further processing is often required to reduce the size and level of noise in images. Having too many features (bands) not only leads to longer executions times in classification, but can negatively affect classification accuracies when there are too few training data and too many features [45]. Therefore, feature reduction or selection is often performed prior to the classification of remotely sensed images. This is especially crucial when using hyperspectral imagery that potentially contain hundreds of bands. Another crucial preprocessing step is the removal of noise, essential when using hyperspectral imagery that frequently contains a nontrivial amount of noise. Too much noise will negatively affect classification accuracies [59].

The focus of this thesis is to develop and analyze an efficient family of automated classification algorithms, the iterative guided spectral class rejection (IGSCR) and continuous iterative guided spectral class rejection (CIGSCR) classification algorithms, that are capable of producing accurate hard and soft classifications. As data reduction and noise removal are so essential to producing accurate classifications, a significant portion of this thesis is devoted to data reduction methods and noise removal methods, namely feature reduction using the singular value decomposition (SVD) and feature reduction and noise removal using the maximum noise fraction (MNF).

PIGSCR was developed to produce fast and portable code using Fortran 95, OpenMP, and the Hierarchical Data Format version 5 (HDF5) and accompanying data access library. The applicability of the faster pIGSCR algorithm is demonstrated by classifying Landsat data covering most of Virginia, USA into forest and non-forest classes with approximately 90 percent accuracy. Parallel results are given using the SGI Altix 3300 shared memory computer and the SGI Altix 3700 with as many as 64 processors reaching speedups of almost 77. This fast algorithm allows an analyst to perform and assess multiple classifications to refine parameters. As an example, pIGSCR was used for a factorial analysis consisting of 42 classifications of a 1.2 gigabyte image to select the number of initial classes (70) and class purity (70%) used for the remaining two images.

A feature selection or reduction method may be appropriate for a specific classification method depending on the properties and training required for the classification method, or an alternative band selection method may be derived based on the classification method itself. This thesis introduces a feature reduction method based on the singular value decomposition (SVD). This feature reduction technique was applied to training data from two multitemporal datasets of Landsat TM/ETM+ imagery acquired over a forested area in Virginia, USA and Rondônia, Brazil. Subsequent parallel iterative guided spectral class rejection (PIGSCR) forest/nonforest classifications were performed to determine the quality of the feature reduction. The classifications of the Virginia data were five times faster using SVD based feature reduction without affecting the classification accuracy.

This thesis describes a new algorithm used to adaptively filter a remote sensing dataset based on signal-to-noise ratios (SNRs) once the maximum noise fraction (MNF) has been applied. This algorithm uses Hermite splines to calculate the approximate area underneath the SNR curve as a function of band number, and that area is used to place bands into "bins" with other bands having similar SNRs. A median filter with a variable sized kernel is then applied to each band, with the same size kernel used for each band in a particular bin. The proposed adaptive filters are applied to a hyperspectral image generated by the AVIRIS sensor, and results are given for the identification of three different pine species located within the study area. The adaptive filtering scheme improves image quality as shown by estimated SNRs. Classification accuracies of three pine species improved by more than 10% in the study area compared to that achieved by the same discriminant method without adaptive spatial filtering.

This thesis introduces the continuous iterative guided spectral class rejection (CIGSCR) classification method based on the iterative guided spectral class rejection (IGSCR) classification method for remotely sensed data. Both CIGSCR and IGSCR use semisupervised clustering to locate clusters that are associated with classes in a classification scheme. This type of semisupervised classification method is particularly useful in remote sensing where datasets are large, training data are difficult to acquire, and clustering makes the identification of subclasses adequate for training purposes less difficult. In CIGSCR and IGSCR, training data are used to evaluate the strength of the association between a particular cluster and a class, and a statistical hypothesis test is used to determine which clusters should be associated with a class and used for classification and which clusters should be rejected and possibly refined. In both algorithms, there is an iterative framework that attempts to produce associated clusters from previously rejected clusters. IGSCR uses discrete clustering to produce discrete (hard) classifications, and CIGSCR uses soft clustering to produce soft classifications. The hypothesis test and iteration within IGSCR assume that cluster assignments are discrete, and therefore new continuous versions of both the hypothesis test and the iteration are developed for CIGSCR. Experimental results indicate that the soft classification output by CIGSCR is reasonably accurate (when compared to IGSCR), and the fundamental algorithmic changes in CIGSCR (from IGSCR) result in CIGSCR being less sensitive to input parameters that influence iterations. Furthermore, evidence is presented to show that the semisupervised clustering in CIGSCR produces more accurate classifications than classification based on clustering without supervision.

The outline of this thesis is as follows. Section I (Chapters 2–8) describes IGSCR and PIGSCR, Section II (Chapters 9–15) introduces SVD-based feature reduction, Section III (Chapters 16–23) provides a full description of the adaptive noise filter, and Section IV (Chapters 24–33) details CIGSCR. Chapter 34 concludes the thesis.

Section I. PIGSCR.

Chapter 2: PIGSCR INTRODUCTION

As remote sensing datasets continue to increase in size, there is a demonstrated need for faster computing resources to decrease processing time. Furthermore, when dealing with certain classification algorithms, more accurate results may be obtained by using slightly different input parameters, such as the number of clusters for K-means [41]. A significantly faster (parallel) implementation of these classification algorithms would allow an analyst to make several runs using different parameters in the equivalent time required to make one serial run, potentially producing more accurate classification results. Although there are increasingly more parallel computers available to the research community, porting existing serial applications to a parallel environment is usually nontrivial.

This thesis discusses specific changes that are made to the IGSCR (iterative guided spectral class rejection) classification algorithm to produce a shared memory parallel algorithm (PIGSCR) with accompanying pseudocode for the classification algorithms [95][67]. A further goal of this implementation is to create source code that is both portable and open source. The final PIGSCR code runs on multiple hardware platforms and operating systems, and it does not have the same "black box" that is associated with commercial software libraries. A further goal of this work is to demonstrate the utility of the PIGSCR implementation by accurately and efficiently classifying Landsat data covering the state of Virginia into forest and non-forest land use informational classes.

Chapter 3: PIGSCR LITERATURE REVIEW

3.1 IGSCR

Unsupervised classification is a process by which all pixels or objects with similar spectral values (spectral classes) are identified (clustering) and then subsequently labeled with respect to informational classes (labeling). Supervised classification, in contrast, requires analyst identification of the spectral classes within each informational class beforehand (training). Remaining pixels or objects are then assigned to a spectral class using a decision rule (classification). As with unsupervised classification, the resulting map must be labeled with respect to informational classes, but for supervised classifications this is trivial since the informational class to which each spectral class belongs was identified in the training stage.

IGSCR is an example of a hybrid classification method, a classification method that exhibits characteristics of both unsupervised and supervised classification [77]. Hybrid classification methods combine multiple classifiers to reduce the workload of the human analyst, most often in the training phase. Bruzzone and Prieto [16] use unsupervised classification (clustering) to modify the spectral signatures generated from a supervised classification so the same training data can be used on images of the same landscape acquired on different dates. Byeungwoo and Landgrebe [18] use a hybrid approach to create a one class classification where the analyst need only train for the class of interest and then unsupervised classification is used to generate signatures for a supervised classification of the original image. Guided clustering requires a user to select training data to represent predefined informational classes, automatically identifies spectral classes (clusters of pixels with similar brightness value vectors) within each informational class (category, such as deciduous forest or row crops) using a clustering algorithm, and then uses the resulting spectral class signatures to perform a supervised classification [5]. This method is advantageous because accurate results are produced while allowing for a greater amount of automation. Guided clustering cannot be entirely automated, however, as user interaction is required after the training process to oversee spectral class creation and refine parameters. IGSCR is more disposed to automation as no user interaction is required after the training phase [95]. IGSCR uses a process called "cluster busting" first introduced by Jensen et al. [47] to refine spectral classes iteratively prior to application of a decision rule. Each spectral class produced by clustering is assigned the value of the majority informational class if that spectral class is statistically determined to be sufficiently pure. In practice, IGSCR-derived area estimates were shown to exceed established precision standards in the USDA Forest Service Forest Inventory and Analysis (FIA) program [67]. IGSCR was also used recently for a study on Sudden Oak Death where Kelly et al. [55] demonstrated that the IGSCR hybrid classification method outperformed both supervised and unsupervised methods alone.

3.1.1 Automation

The IGSCR algorithm requires a clustering algorithm and a means by which brightness value vectors are assigned to clusters (the decision rule), so the choice of the specific algorithms that are used may be dependent on implementations that are available. In 2003, IGSCR was implemented in the C language using a commercial remote sensing image processing package [67]. The decision rule in this prior implementation is maximum likelihood [77]and the clustering algorithm is ISODATA [3].

3.2 Parallel Computing

The first distributed memory parallel computers were both expensive and difficult to use, making them inaccessible to the general scientific community, although vector parallel computers (CDC and Cray) quickly became the norm in scientific computing [75]. The emergence of commodity clusters has reduced the startup cost required to build a very powerful computer containing thousands of processors. Even specialized shared memory multicomputers containing a modest number of processors have become a viable, affordable option for scientific researchers.

To coincide with more publicly available hardware solutions, two software solutions have emerged as standards for programming on their respective architectures. MPI (Message Passing Interface) is a standard describing directives that extend an already existing programming language such as C or Fortran [87]. MPI is the protocol of choice for a distributed memory parallel computer because communication between processors using MPI is done explicitly.

For shared memory programming, OpenMP is the standard that is commonly used [50]. OpenMP uses a fork/join model of parallelization and is an extension to an existing programming language comprised of compiler directives and function calls [50]. OpenMP directives are formed such that the underlying serial code is not perturbed and will therefore still compile without using OpenMP. Using OpenMP to parallelize applications on a shared memory computer allows a programmer to specify serial and parallel portions of code while hiding communication and memory access details.

3.2.1 Comparison of OpenMP and MPI

Although MPI can be used on a shared memory computer, it is more advantageous to develop parallel code using OpenMP on a shared memory platform. First, OpenMP does not require explicit communication between processors because all processors have access to one large global memory. Data transfer is completely contained in the underlying memory management hardware and is not exposed to the programmer. A second advantage resulting from a single memory space is having the luxury of developing parallel code incrementally from the serial code [75]. For example, starting with a serial program it is possible to separate different parts of code into blocks and parallelize one block at a time. With the fork/join model, the benefit of partial parallelization is realized using multiple processors while the serial portions of code are run using just one processor. The main disadvantage associated with OpenMP is that it is effectively limited to shared memory programming and is not (yet) suitable on a distributed memory platform. If the speedup (execution time on multiple processors divided by the best execution time on one processor) required is much larger than the number of processors available on a shared memory machine, a large distributed memory supercomputer and MPI should be used.

Chapter 4: CLASSIFICATION ALGORITHMS

4.1 K-means

K-means clusters N data points in E^b (real b-dimensional Euclidean space, all b-tuples of real numbers) into K different clusters such that the sum of the distances between each cluster mean and the data points belonging to the cluster cannot be reduced by reassigning any of the data points to any of the existing clusters (Hartigan, 1975). The algorithm begins with K initialized cluster mean points $m^{(1)}, \ldots, m^{(K)} \in E^b$, and each data point $x^{(i)} \in E^b$ is assigned to the nearest cluster mean according to a distance measure such as Euclidean distance,

$$\|x - m\|_2 = \sqrt{\sum_{j=1}^{b} (x_j - m_j)^2}.$$

Once all data points have been assigned, cluster means are recomputed by

$$m^{(k)} = \frac{1}{N_k} \sum_{i \in I_k} x^{(i)}$$

where $m^{(k)}$ is the new mean for cluster k, I_k is the set of indices of points assigned to cluster k, and $N_k = |I_k|$. The index sets $I_1, \ldots, I_K$ constitute a partition of $\{1, \ldots, N\}$, and are assumed to be nonempty. The assign and recompute process iterates until there is no significant change in cluster assignment, determined by a user set threshold of number of data values reassigned or total distance the means migrated between iterations.

K-means will converge in a finite number of steps to a local minimum point of the objective function

$$\sum_{k=1}^{K} \sum_{i \in I_k} \left\| x^{(i)} - m^{(k)} \right\|_2^2 .$$

There is no guarantee or expectation that this local minimum point is equal to or even close to the global minimum point (Hartigan, 1975). There have been several methods proposed for initializing the cluster means in order to produce a good clustering, which will be discussed in a later chapter. The K-means algorithm is given below.

```
Algorithm Kmeans(its, x, threshold, K, cluster, sigs)
Input: its (maximum number of iterations)
x (three-dimensional image)
threshold (iteration termination criteria)
K (number of clusters)
Output: cluster (two-dimensional cluster assignment array
for x)
sigs (cluster signatures)
begin
  call Initialize_Means(x, K, m); (where $m^{(1)}, \ldots, m^{(K)}$
are the means for clusters $1, \ldots, K$)
  for it := 1 step 1 until its do
```

```
    begin
      for $k := 1$ step 1 until $K$ do
        begin
          $sum^{(k)} := 0$;
          $n_k := 0$;
        end
      $pixels\_changed := 0$; (initialize number of pixels
      that change assignment
      to zero)
      for $i := 1$ step 1 until $rows$ do
        for $j := 1$ step 1 until $cols$ do
          begin
            $min\_distance := large\_number$; (initialize
            minimum distance squared)
            $min\_c := 0$; (nearest cluster index)
            for $k := 1$ step 1 until $K$ do
              begin
                $distance := \sum_{p=1}^{b} (x_p^{(i,j)} - m_p^{(k)})^2$;
                if ($distance < min\_distance$) then
                  begin
                    $min\_c := k$;
                    $min\_distance := distance$;
                  end
              end
            $sum^{(min\_c)} := sum^{(min\_c)} + x^{(i,j)}$;
            $n_{min\_c} := n_{min\_c} + 1$;
            if ($min\_c \neq cluster_{i,j}$) then
              begin
                $cluster_{i,j} := min\_c$;
                $pixels\_changed := pixels\_changed + 1$;
              end
          end
      for $k := 1$ step 1 until $K$ do
        if ($n_k \neq 0$) then
          $m^{(k)} := sum^{(k)}/n_k$;
        else
          Delete Cluster $k$;
      if ($pixels\_changed/no\_of\_pixels < threshold$) then
        exit loop;
    end
  compute class statistics for cluster signatures $sigs$;
end
```

4.1.1 Implementation

Because of the vector operations that are implicit with an implementation of K-means, there are numerous opportunities to exploit the use of Fortran 95 intrinsic functions for both serial and parallel implementations [33]. During an iteration of K-means where each pixel is being assigned to a cluster, the cluster assignment can be stored in an array with dimensions equal to that of the two-dimensional image. Element (i, j) of the cluster assignment array would correspond to the (i, j) element, a brightness vector, of the image array. This classification array is not only the output classification image for K-means, but it is also a mechanism for elegantly computing certain cluster statistics after the iteration terminates. A running tally of the sum and number of data values for each cluster is computed during each iteration in order to compute each cluster mean upon completion of cluster assignment for the entire image. Since the operator '/' is defined for arrays in Fortran 95, all the cluster means are elegantly computed by dividing the sum vector by the previously calculated number of data values `N(k)` via

```
mean(1:b, k) = sum(1:b, k) / N(k),
```

where `mean` is a two-dimensional array containing mean vectors for each cluster, `sum` is the sum of the data values belonging to cluster `k`, and `b` is the number of bands in the image.

Once the iteration has converged, there are additional statistics calculated as part of the output signature of each cluster. These include minimum and maximum vectors, and a covariance matrix for each cluster. The covariance matrix calculation will be described in greater detail in a later chapter. In order to calculate the minimum and maximum, an array containing only data values belonging to a specific cluster is created using the Fortran 95 function `PACK`. The cluster assignment array may be used to form a masked array that will be input into the function `PACK` to produce a one-dimensional array that can be input into an array intrinsic function. For example, for a specific cluster and a specific band, the minimum value is found by masking out all values not associated with the cluster of interest from the two-dimensional array of data values for the band of interest, and then the resulting array is the input to `PACK`. The resulting output is a one-dimensional array containing all data values belonging to the cluster of interest for one band, and this will be the input array for the Fortran 95 intrinsic function `MINVAL`, which will return the minimum value. An example of the use of the above functions is

```
pArray(1:N(k)) = PACK(class(1:r, 1:c, i), class(1:r,1:c, i).eq.k)
          min(i) = MINVAL(pArray(1:N(k))),
```

where min is a one-dimensional array of length b, `pArray` is the one-dimensional result of the call to `PACK`, `i` is the band of interest, `N(k)` is the number of data values attributed to the cluster `k` for which the statistics are being computed, and `r` and `c` are the number of rows and columns, respectively. The maximum is found using the Fortran 95 intrinsic function `MAXVAL`, which is analogous to the function `MINVAL` described above.

4.2 K-means Initialization

Principal components analysis (PCA) or Karhunen-Loeve analysis transforms a highly correlated dataset into an uncorrelated dataset where the first principal component (PC1) accounts for the most variance in the original multi-dimensional dataset [48]. PCA is often used to reduce the dimensionality of a dataset without losing much information [48].

There are numerous ways to determine initialized mean placement for K-means, including randomly seeding the means [41]. As an improvement, Hartigan [41] suggests that the means be seeded along the axis of greatest variance (PC1), and Huang [44] showed that placing the initial means plus or minus one standard deviation from the mean results in higher classification accuracies. The algorithm for K-means initialization using the first principal component axis to seed the cluster means is given below. For a matrix A, $A_{i\cdot}$ denotes the ith row, $A_{\cdot i}$ denotes the ith column, and A^t is the transpose.

```
*Algorithm Initialize_Means(x, K, m)*
**Input:** $x$ (three-dimensional image)
$K$ (number of cluster means to initialize)
**Input/Output:** $m$ ($K$ cluster mean vectors)
**begin**
  $\Sigma := \frac{1}{rows * cols - 1} \sum_{i=1}^{rows} \sum_{j=1}^{cols} (x^{(i,j)} - m)(x^{(i,j)} - m)^t$
  (where $\Sigma$ is the covariance matrix and $rows$ and $cols$
  are the number of rows and columns in $x$)
  factor $\Sigma = PDP^t$ (where $D$ is a diagonal matrix
  containing the eigenvalues of $\Sigma$ and $P$ contains
  the corresponding orthonormal eigenvectors of $\Sigma$)
  **for** $i := 1$ **step** 1 **until** $rows$ **do**
    **for** $j := 1$ **step** 1 **until** $cols$ **do**
      $PC1_{i,j,1} := P_{\cdot 1}^t x^{(i,j)}$; (where $PC1$ is the principal
      component image)
  $\mu_{PC1} := P_{\cdot 1} 1^t m$
  $\sigma_{PC1} := D_{1,1}$ (The variance is the largest eigenvalue
  distribute $\mu_1, \ldots, \mu_K$ evenly between $\mu_{PC1} - \sigma_{PC1}$
  and $\mu_{PC1} + \sigma_{PC1}$;
  **for** $i := 1$ **step** 1 **until** $rows$ **do**
    **for** $j := 1$ **step** 1 **until** $cols$ **do**
      $class_{i,j} := k$ where $|\mu_k - PC1_{i,j,1}| =$
      $\min_{l \in K} |\mu_l - PC1_{i,j,1}|$;
  recompute class means;
**end**
```

4.2.1 Implementation

The PCA calculation first requires a covariance matrix

$$\Sigma = \frac{1}{N-1} \sum_{i=1}^{N} (x^{(i)} - m)(x^{(i)} - m)^t$$

where $x^{(1)}, \ldots, x^{(N)} \in E^b$ are b-dimensional data points, m is the sample mean, and N is the total number of data points. This formula for the covariance matrix requires a previously calculated mean and would require two iterations through the data. Consider the i, j element of $(N-1)\Sigma$,

$$\begin{aligned}(N-1)\Sigma_{i,j} &= \sum_{k=1}^{N}(x_i^{(k)} - m_i)(x_j^{(k)} - m_j) \\ &= \sum_{k=1}^{N} x_i^{(k)}x_j^{(k)} - x_i^{(k)}m_j - x_j^{(k)}m_i + m_im_j \\ &= A_{i,j} - s_im_j - s_jm_i + Nm_im_j,\end{aligned}$$

where

$$s = \sum_{k=1}^{N} x^{(k)},$$

$$A = \sum_{k=1}^{N} x^{(k)}x^{(k)t},$$

and

$$m = \frac{s}{N}$$

can be calculated in one iteration through the data points $x^{(1)}, \ldots, x^{(N)}$. To save computation time and storage space, only half of Σ and A need be stored and calculated as they are both symmetric. To calculate the eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_b \geq 0$ and corresponding eigenvectors $v^{(1)}, \ldots, v^{(b)}$, where b is the number of bands of the data values $x^{(1)}, \ldots, x^{(N)}$, only one LAPACK routine `SSYEV` (or its double precision equivalent `DSYEV`) is required [2]. Once these are calculated, the brightness vectors $x^{(1)}, \ldots, x^{(N)}$ and the mean m may be transformed to PC1 using the transformation

$$PC1_k = v^{(1)t}x^{(k)}, \quad \mu_{PC1} = v^{(1)t}m, \quad k = 1, \ldots, N,$$

where $v^{(1)}$ is the eigenvector corresponding to the largest eigenvalue λ_1. Once the mean and all brightness values are transformed to PC1, K cluster means are initialized equally spaced in the interval $[\mu_{PC1} - \lambda_1, \mu_{PC1} + \lambda_1]$, and each brightness vector is assigned to the nearest cluster based on its first principal component. Once all vectors have been assigned to a cluster, each cluster mean is calculated using each pixel's non-transformed brightness vector.

4.3 Maximum Likelihood

Maximum likelihood is a highly used decision rule [77]. The algorithm determines the class to which a pixel belongs based on the probability that it belongs to each of the input training classes $\omega_1, \ldots, \omega_t$. The following derivation of the maximum likelihood decision rule is taken from [77]. Maximum likelihood is based on the Bayes decision rule where a point $x \in \omega_i$ if $p(\omega_i|x) > p(\omega_j|x)$ for all $j \neq i$. The probabilities $p(\omega_1|x), \ldots, p(\omega_t|x)$ are rarely known at classification time, but through training $p(x|\omega_i)$ can be estimated, and $p(\omega_i|x) = p(x|\omega_i)p(\omega_i)/p(x)$. For a pixel x, removing the common term $p(x)$:

$$x \in \omega_i \text{ if } p(x|\omega_i)p(\omega_i) > p(x|\omega_j)p(\omega_j) \text{ for all } j \neq i.$$

Because a logarithm function is monotonically increasing, it follows that

$$\ln(p(x|\omega_i)p(\omega_i)) > \ln(p(x|\omega_j)p(\omega_j))$$

for all $j \neq i$ implies $p(x|\omega_i)p(\omega_i) > p(x|\omega_j)p(\omega_j)$ for all $j \neq i$ (Duda and Hart, 1973). Assume a Gaussian distribution for the training sets,

$$p(x|\omega_i) = (2\pi)^{\frac{-b}{2}}|\Sigma_i|^{-1/2}e^{-\frac{1}{2}(x-m^{(i)})^t\Sigma_i^{-1}(x-m^{(i)})},$$

where b is the number of bands, Σ_i is the covariance matrix for class i, $m^{(i)}$ is the mean for class i, and $|\Sigma_i|$ is the determinant of the covariance matrix. It is assumed that $|\Sigma_i| \neq 0$. Since $\ln(p(x|\omega_i)p(\omega_i))$ is being maximized, any term not dependent on i can be removed, and if no prior knowledge of $p(\omega_i)$ is known, equal prior probabilities are assumed for all classes. After removing non-discriminating terms, the function to be maximized (with respect to i) is

$$g_i(x) = -\ln|\Sigma_i| - (x - m^{(i)})^t\Sigma_i^{-1}(x - m^{(i)}).$$

The maximum likelihood algorithm is given below.

```
*Algorithm Maximum_Likelihood(classes, x, class)*
**Input:** *classes* (class signatures containing covariance
matrix and mean)
$x$ (three-dimensional image)
**Output:** *class* (classification of $x$)
**begin**
  **for** $k := 1$ **step** 1 **until** $|classes|$ **do**
    **begin**
      diagonalize $\Sigma_k$ to produce $PDP^t$; ($\Sigma_k$ is the
      covariance matrix for class $k$, $D$ is a diagonal matrix
      containing the eigenvalues of $\Sigma_k$, and $P$ contains
      orthonormal eigenvectors corresponding to $D$)
      $|\Sigma_k| := \Pi_{j=1}^{b}(D_{jj})$; (calculate determinant of $\Sigma_k$
      where $b$ is the number of bands)
      $\Sigma_k^{-1} := PD^{-1}P^t$
    **end**
  **for** $i := 1$ **step** 1 **until** *rows* **do**
  for $j := 1$ step 1 until $cols$ do
    begin
      $max\_G := -large\_number$; ($large\_number$ is the
      largest possible value of $max\_G$)
      $max\_class := 0$; (the highest probability index)
      for $k := 1$ step 1 until $|classes|$ do
        begin
          $g := -\ln|\Sigma_k| -$
          $(x^{(i,j)} - m^{(k)})^t \Sigma_k^{-1} (x^{(i,j)} - m^{(k)})$;
          if $(g > max\_G)$ then
            begin
              $max\_G := g$;
              $max\_class := k$;
            end
        end
      $class_{i,j} := max\_class$;
    end
end
```

4.3.1 Implementation

In this maximum likelihood implementation, it is assumed that mean vectors and covariance matrices will be input as part of spectral class signatures. The goal of this implementation is to efficiently and portably code the above decision rule.

A quick glance at the above equation reveals that the most costly computations are calculating the determinant and inverse of the covariance matrix. Because the covariance matrix is symmetric, it may be factored as $\Sigma = PDP^{-1}$, where the columns of P are the orthonormal eigenvectors $u^{(1)}, \ldots, u^{(b)}$ and D is a diagonal matrix containing eigenvalues $\lambda_1, \ldots, \lambda_b$, corresponding to the eigenvectors [62]. The determinant of Σ is the determinant of D, and the determinant of a triangular matrix is the product of the elements of the diagonal [62]. To calculate the inverse of Σ, the inverse of PDP^{-1} may be calculated as

$$(PDP^{-1})^{-1} = (P^{-1})^{-1}(D)^{-1}(P)^{-1} = PD^{-1}P^{-1}.$$

$P^{-1} = P^t$ because the eigenvectors are orthonormal and D^{-1} is trivial to calculate because only the inverse of each scalar term in the matrix diagonal need be calculated.

In order to implement this efficiently, the LAPACK function `SSYEV` (`DSYEV` for double precision) is used to calculate the eigenvectors and eigenvalues of Σ [2]. After determining P and D, Fortran 95 intrinsic functions `MATMUL` and `TRANSPOSE` maybe be used to complete the inverse calculation.

4.4 IGSCR

The algorithm begins with the user inputting an image to be classified, training data specific to that image, and various parameters for the unsupervised classification and the spectral class homogeneity test. K-means clustering is performed on the input image, and the resulting spectral classes are analyzed for informational class homogeneity to ensure that only one informational class is present (with high probability) in each spectral class. If more than one informational class is present with significant probability, that spectral class will be rejected, and all pixels belonging to that class will have the opportunity to be reclustered. In order to conduct the homogeneity test, the training data must be analyzed to determine how many points from each informational class fall into each spectral class. For any training point, t_i, the point's coordinates in the original image are used to determine which spectral class it was assigned to in the unsupervised classification. Each spectral class has an informational class count vector that has a length equal to the number of informational classes, and the component of the vector that corresponds to the informational class that was associated to t_i (by the analyst in the training phase) is incremented. After this process has been applied to each training point, the homogeneity test is conducted for each spectral class. The homogeneity test is given by Musy et al. [67] as

$$N_k(1-p_0) \geq 5 \text{ (spectral class must have minimum number of pixels) and } Z > Z(\alpha) \text{ (homogeneity test),}$$

where

Z	$= (\hat{p} - p_0 - 0.5/N_k)/\sqrt{p_0(1-p_0)/N_k}$,
α	is the type-I error rate,
$Z(\alpha)$	is the value such that $P(Z \geq Z(\alpha)) = \alpha$ for a $N(0,1)$ variable Z,
$\hat{p}$	$= N_{maj}/N_k$,
p_0	is the user input threshold,
N_k	is the total number of training pixels in the spectral class, and
N_{maj}	is the majority informational class count (count of training pixels belonging to the informational class with the highest count).

If a spectral class is determined to be pure ($Z > Z(\alpha)$), then the majority informational class is assigned to the spectral class, and each pixel that was assigned to the spectral class by K-means clustering is recoded to the informational class value in the output classification image. Furthermore, each pixel belonging to a pure spectral class is removed from the input image for successive applications of K-means clustering. The pure spectral class and its signature (a set of statistics for the class such as mean and covariance among bands) are kept for later processing. After all spectral classes have been processed, K-means clustering (with the same value of K that was used in each previous iteration) is used again on the remaining pixels and the homogeneity test ensues. This iteration continues until no pixels belonging to impure classes remain in the input image, no pure classes were found during the previous iteration, or a maximum number of iterations has occurred. In this manner, all pixels belonging to impure spectral classes are reclustered during the subsequent iteration and each iteration operates on a proper subset of the pixels used in the previous iteration.

Once the iteration terminates, there is a set of pure spectral classes, an unsupervised classification image recoded to informational class values with zeroes for background and unclassified pixels, a set of unclassified pixels remaining from the original image, and the original image. There are three options for output classification images at this stage, and any or all of them may be selected by the user. First, a maximum likelihood classification may be run on the original image using the pure spectral classes as training classes, producing a Decision Rule (DR) image. Second, all unclassified pixels may be recoded to a reserved value designating unclassified pixels (the number of informational classes plus one) and added to the unsupervised classification image to produce an Iterative Stacked (IS) image. Finally a maximum likelihood classification may be performed on only the unclassified pixels (using the pure spectral classes as training classes), and the resulting classification may be added to the unsupervised classification image to produce a Iterative Stacked plus (IS+) image. The IGSCR algorithm is given below.

```
*Algorithm IGSCR(image, P, DR, IS, IS+, $\alpha$, $p_0$, its,
classes, kits, threshold, sigs, DR_image, IS_image,
IS+_image)*
**Input:** *image* (three-dimensional image)
$T$ (set of training data containing x,y coords and an
informational class value)
*DR, IS, IS+* (Boolean values corr. to output)
$\alpha$ (type-I error rate)
$p_0$ (homogeneity threshold)
*its* (number of iterations for main loop)
*classes* (number of classes to create in $K$-means loop)
*kits* (number of $K$-means iterations)
*threshold* (loop-ending criteria for $K$-means)
**Output:** *sigs* (set of pure signatures)
*DR_image* (image resulting from maximum likelihood
on input image)
*IS+_image* (image resulting from maximum likelihood
on impure pixels)
*IS_image* (image resulting from impure pixels being
recoded in unsupervised image)
**begin**
  *k_image := image* (*k_image* is input to each $K$-means call)
  **for** $i := 1$ **step** 1 **until** *its* **do**
    **begin**
      **if** (all pixels in *k_image* are zero) **then**
        exit loop;
      *call Kmeans(kits, k_image, threshold, classes,
      k_classimage, ksigs)*;
      reset spectral class counts;
      *pure_classes* := 0; (number of pure classes)
      **for** $j := 1$ **step** 1 **until** $|T|$ **do**
        **begin**
          $c := k\_classimage(T_x^{(j)}, T_y^{(j)})$;
```

```
                if ($c \neq 0$) then
                    increment the informational class count for
                    class of $P^{(j)}$ and total count within spectral
                    class $c$;
            end
        for $j = 1$ step 1 until $|ksigs|$ do
            begin
                if ($(total_j(1 - p_0)) \geq 5$) then (where $total_j$
                represents the count of pixels belonging to $j$)
                    begin
                        Determine $N_{maj}$ and $C_{maj}$ where $N_{maj}$ is
                        the no. of pixels in maj. info. class, $C_{maj}$
                        $\hat{p} := \frac{N_{maj}}{total_j}$;
                        $Z := \frac{(\hat{p} - p_0 - .5/total_j)}{\sqrt{p_0(1-p_0)/total_j}}$;
                        if ($Z > Z(\alpha)$) then
                            begin
                                increment pure_classes;
                                add $ksig(j)$ to group of sigs;
                                mask $j$ from k_image;
                                recode where k_classimage $= j$ to
                                $C_{maj}$ in class_image;
                                (where class_image is the stacked
                                output of the $K$-means classifications)
                            end
                    end
            end
        if (pure_classes $= 0$) then
            exit loop;
    end
  if ($DR$) then
    call Maximum_Likelihood(sigs, image, DR_image);
  if ($IS+$) then
    begin
      call Maximum_Likelihood(sigs, k_image,
      IS+_image);
      IS+_image := IS+_image + class_image;
    end
  if ($IS$) then
    begin
      recode k_image non-zeros to number of informational
      classes + 1, store result in k_image;
      IS_image := k_image + class_image;
    end
end
```

4.4.1 Implementation

The biggest opportunities to exploit the intrinsic functions in Fortran 95 are in the masking and recoding phase. Without vector operations, it would be necessary to loop over each pixel and test the pixel's class value explicitly. Based on this test, the pixel's brightness vector from the input image would either be masked out of the input image or recoded to its informational class value in the output classification image. Using the Fortran 95 intrinsic function `WHERE` with an array mask, an array can be used as a Boolean test to determine whether an operation should be performed on corresponding elements of another array of equal dimensions. An example of using the `WHERE` statement to recode to informational class values is

```
WHERE(class.eq.k)
recodeImage = classValue(k)
end WHERE,
```

where the dimensions of `class` and `recodeImage` are equal. A similar block is used to mask the pure pixels out of the input image. Another subtle opportunity for improved performance with vector statements arises where it is necessary to use matrix addition to add two images together, such as when the result of a maximum likelihood classification is added to the stacked result of the iterative unsupervised classifications.

Chapter 5: HIERARCHICAL DATA FORMAT 5 (HDF5)

One goal of this implementation is to achieve portability by removing dependencies on specific platforms and software, which would limit the use of the proprietary data format that was used in the previous implementation. The HDF5 data format and API library was chosen because it is flexible, robust, and already widely used in the scientific community [43]. HDF5 can be used on a variety of operating systems with many C and Fortran compilers [43].

HDF5 is a standard that defines a grouping of large scientific datasets and associated properties stored on disk, and it is an API that implements the standard in order to provide efficiency and ease to a programmer. There are various objects defined in HDF5 that can be grouped together in a conceptual tree structure within an HDF5 file. Reading and writing of each object is done through a series of function calls. Table 1 describes all of the HDF5 objects that are used in the implementation of PIGSCR. These objects are all that is necessary to describe the files that are input into and output from PIGSCR.

Table 1. HDF5 Objects.

file	container to store file objects
group	acts like unix directory to form hierarchy of objects within file
dataset	large array objects used to store multi-dimensional data
dataspace	describes dimensionality of a dataset
datatype	can be intrinsic or derived type; description of data in a dataset
attribute	generally a small dataset (can be scalar) used to describe metadata for a user-defined property of a group, dataset, or a named datatype

5.1 Image File

The input image file contains a three-dimensional dataset containing the image with a corresponding dataspace and datatype. The three dimensions correspond to rows, columns, and bands of the image. Additionally, there are multiple attributes used to specify information such as corner map coordinates, pixel size, and projection information for the image. This information corresponds to data that would typically be found in a header file accompanying binary data. Although this information is not used by PIGSCR, it may be useful to the analyst in a later application, and therefore it is retained. In order for PIGSCR or any program to extract the image data, it uses the HDF5 API to access the dataspace and datatype to determine the size of the image and precision of each element. This information is used in one function call to directly read the conceptual three-dimensional image on disk into a three-dimensional array in memory. There are equivalent function calls to write an image to file, which are used to output the final two-dimensional classification images.

5.2 Training Data Files

A single training data file is input into PIGSCR. The training data is represented by a set of x, y coordinates and the corresponding informational class values for the set of points. The x and y coordinates are file indices for specific points in the input image. Each set of training points is a two-dimensional dataset where the first dimension has a length of three (for two coordinates and a class value) and the length of the second dimension is the number of points.

5.3 Signature Files

PIGSCR outputs the pure class signatures that were used for the supervised classification to an HDF5 file. Each class signature is represented by an HDF5 group with the class name as the group name. Each signature group contains the following datasets: minimum, maximum, mean, covariance matrix, and n, a scalar value representing the total number of points in the class. Each signature file has an attribute named "classes" specifying the total number of class signatures contained in the file.

5.4 Utility Programs

As stated above, training data and images are input into PIGSCR in the HDF5 format. Two utilities were written to create these two types of files. One utility converts generic binary images with header files to HDF5 image files as described above. A second utility was written to convert three-column ASCII files containing file or map coordinates and class values to one HDF5 training data file. There is an HDF5 utility called h5dump that will allow a user to examine the contents of any HDF5 file, which is useful for examining the contents of a signature file.

Chapter 6: PIGSCR

The strategy for parallelizing IGSCR is to locate operations in the original algorithm that may be run in parallel and to concentrate on those operations that will result in the greatest speedup. These operations that may be run concurrently will fall into one of two categories of parallelism, functional or data parallelism. Data parallelism, or single instruction multiple data (SIMD) parallelism, is exhibited when multiple processes perform identical operations on different members of one dataset, whereas functional parallelism, or multiple instruction multiple data (MIMD) parallelism, occurs when distinctly different operations are performed concurrently on potentially independent data [75]. An example of functional parallelism inherent in IGSCR is when three different operations may be applied to the input image to produce three independent classifications. Data parallelism is prevalent throughout IGSCR; a simple example of this is the maximum likelihood classification where all pixels are classified using identical operations. Loops over many data values are common indicators of potential data parallelism.

In order to isolate those sections of code that may be run in parallel with a modest shared memory programming adaptation, it is necessary to determine which parts of the sequential algorithm are independent. The following three conditions, known as Bernstein's conditions, guarantee that two statements S_i and S_j are independent:

$$O(S_i) \cap O(S_j) = \emptyset,$$
$$I(S_j) \cap O(S_i) = \emptyset,$$
$$I(S_i) \cap O(S_j) = \emptyset,$$

where the set of all input, output variables to statement S_i is denoted by $I(S_i)$, $O(S_i)$ respectively [9]. Simply put, two statements may be considered independent if the two statements do not write to the same variable and one statement does not have an output variable that is also an input variable to the other statement.

Once the candidate sections for parallelization are selected, it is necessary to determine whether parallelization will result in an appreciable speedup in the overall algorithm. An example of parallelism that does not best take advantage of potential processing power is the functional parallelism inherent in the multiple classifications at the end of IGSCR (maximum likelihood, stacked unsupervised classification, and stacked unsupervised with maximum likelihood), because there are at most three classification tasks and therefore this (functional) parallelism is not scalable beyond three processors. There are other cases when parallelizing a section of code does not result in a faster algorithm, such as when the overall speedup of a parallel section will be negated by overhead associated with parallel communication. Therefore it is important to locate sections of code that may be run in parallel and have a large potential speedup and high likelihood of contributing to large speedup of the entire algorithm. A profiler is a useful tool to aid an analyst in determining which sections of code might benefit most from parallelization efforts. The output of a profiler will specify the percentage of total time spent in each function within code. By profiling a serial implementation of an algorithm, an analyst may gain insight into which portions of the code are most costly, and the analyst will consequently concentrate most parallelization effort on those sections.

Table 2. Profiling Results.

Function	Percentage of total time
IGSCR	100
K-means	79.5
K-means output calculations	22
Initialize K-means	11.5
Maximum Likelihood	14

Table 2 shows the approximate results of profiling a serial implementation of IGSCR. The most costly functions are listed.

These results show that the majority of time spent running IGSCR is spent running K-means followed by maximum likelihood. A significant reduction in time spent in K-means will significantly reduce the overall time required to run IGSCR. Furthermore the profiling provides further insight into the various functions within K-means that are contributing to the large amount of processing time required. It is interesting to note that the time spent in the IGSCR code (determining class homogeneity and subsequently modifying the input and output images) is relatively small in comparison to the total time required to run its clustering and classification functions.

The most obvious opportunities for parallelization and largest performance gains in IGSCR are the loops that have a large number of predetermined independent iterations. These loops are present throughout IGSCR, most notably in the K-means clustering and maximum likelihood classification algorithms. For clustering, an operation is applied to an individual pixel to determine a cluster assignment, and the calculations for all pixels may be performed concurrently as the operation on each pixel is independent of all other pixels and resulting clustering output. The same is true for the loop to calculate spectral class statistics in maximum likelihood. Because there are far fewer spectral classes than pixels, the speedup may not be as high, but there will still be speedup as a result of calculating class statistics in parallel when many spectral classes are present (enough to compensate for parallel overhead). The shared memory parallel maximum likelihood algorithm therefore differs from the serial version when one process forks to form several processes to calculate class statistics and perform the classification of the image. The parallel maximum likelihood algorithm is given below.

```
Algorithm SIMD_Maximum_Likelihood(classes, x, class)
Input: classes (class signatures containing covariance
matrix and mean)
x (three-dimensional image)
Output: class (classification of x)
begin
  for $k := 1$ step 1 until $|classes|$ fork STATLOOP
  STATLOOP:
    begin
      diagonalize $\Sigma_k$ to produce $PDP^t$; ($\Sigma_k$ is the
      covariance matrix for class $k$, $D$ is a diagonal matrix
      containing the eigenvalues of $\Sigma_k$, and $P$ contains
      orthonormal eigenvectors corresponding to $D$)
```

```
          $|\Sigma_k| := \Pi_{j=1}^{b}(D_{jj})$; (calculate determinant of $\Sigma_k$
          where $b$ is the number of bands)
          $\Sigma_k^{-1} := PD^{-1}P^t$
        **end**
    **join** *no_of_processes*;
    **private** *i, j, max_G, max_class, k, g*
    **shared** *rows, cols, classes,* $\Sigma$*, x, m,* $\Sigma^{-1}$*, class*
    **for** $i := 1$ **step** 1 **until** *rows* **fork** CLASSIFYLOOP
    CLASSIFYLOOP:
        **for** $j := 1$ **step** 1 **until** *cols* **do**
          **begin**
            $max\_G := -large\_number$; ($large\_number$ is the
            largest possible value of $max\_G$)
            $max\_class := 0$; (the highest probability index)
            **for** $k := 1$ **step** 1 **until** $|classes|$ **do**
              **begin**
                $g := -\ln|\Sigma_k| -$
                $(x^{(i,j)} - m^{(k)})^t \Sigma_k^{-1} (x^{(i,j)} - m^{(k)})$;
                **if** $(g > max\_G)$ **then**
                  **begin**
                    $max\_G := g$;
                    $max\_class := k$;
                  **end**
              **end**
            $class_{i,j} := max\_class$;
          **end**
    **join** *no_of_processes*;
**end**
```

In K-means, there are several large loops over all pixels to consider for parallelization. The largest consumer of time is the cluster reassignment loop where each pixel is assigned to the nearest cluster and the cluster mean is recalculated for the subsequent iteration.

In the interest of having a large task granularity, it is desirable to spread the outermost loop over all processes. This reduces the overhead associated with forking multiple processes repeatedly in an inner loop. In this algorithm, the outer loop is arbitrarily over all rows, so all rows will be spread over all processes. The cluster assignment loop is ideal for parallelization because it is large and costly, and each iteration is independent. Each pixel is assigned to the closest cluster, and during a single iteration, no pixel's assignment is dependent on another pixel. At the end of the cluster assignment, the cluster means are determined through the use of running sums and counts that are tallied within the assignment loop. These are arrived at by each process maintaining local running sums and counts, and when the loop terminates all local copies are combined through addition to produce global totals. This is known as a reduction operation.

The other functions within K-means that use a significant amount of time are the functions that initialize the cluster means using principal components analysis and calculate final cluster statistics. In the mean initialization function, there are three loops that iterate over all data values in the input image. These large loops are a good focus for parallelization efforts as they are costly and independent of each other.

First the covariance calculation loop may be broken up across the rows of the image. Recall that the covariance matrix can be calculated in one pass of the image by storing various sums. These sums may be calculated in parallel using a reduction as described above. The second large loop converts each pixel to its corresponding first principal component. This is a straightforward linear transformation that is independent of other pixel values and can rather trivially be done in parallel. Finally, the last large loop assigns each pixel to a cluster based on the distance to the cluster's mean value located on the first principal axis. This loop is conceptually identical to the main K-means cluster assignment loop and can be parallelized in the same manner. The parallel algorithms for initializing the cluster means using principal components and K-means clustering are given below.

```
*Algorithm SIMD_Initialize_Means(x, K, m)*
**Input:** $x$ (three-dimensional image)
$K$ (number of cluster means to initialize)
**Input/Output:** $m$ ($K$ cluster mean vectors)
**begin**
  in parallel: $\Sigma :=$
  $\frac{1}{rows * cols - 1} \sum_{i=1}^{rows} \sum_{j=1}^{cols} (x^{(i,j)} - mean)(x^{(i,j)} - mean)^t$
  (where $\Sigma$ is the covariance matrix and $rows$ and $cols$
  are the number of rows and columns in $x$)
  factor $\Sigma = PDP^t$ (where $D$ is a diagonal matrix
  containing the eigenvalues of $\Sigma$ and $P$ contains
  the corresponding orthonormal eigenvectors of $\Sigma$)
  **private** *i, j*
  **shared** *rows, cols, PC1, P*
  **for** $i := 1$ **step** 1 **until** $rows$ **fork** PCALOOP
  PCALOOP:
    **for** $j := 1$ **step** 1 **until** $cols$ **do**
      $PC1_{i,j,1} := P_{\cdot 1}^t x^{(i,j)}$; (where $PC1$ is the principal
      component image)
  **join** *no_of_processes*;
  $\mu_{PC1} := P_{\cdot 1}{}^t m$
  $\sigma_{PC1} := D_{1,1}$ (The variance is the largest eigenvalue
  distribute $\mu_1, \ldots, \mu_K$ evenly between $\mu_{PC1} - \sigma_{PC1}$
  and $\mu_{PC1} + \sigma_{PC1}$;
  **private** *i, j, k, l*
  **shared** *rows, cols, class, μ, PC1*
  **for** $i := 1$ **step** 1 **until** $rows$ **fork** ASSIGNLOOP
  ASSIGNLOOP:
    **for** $j := 1$ **step** 1 **until** $cols$ **do**
      $class_{i,j} := k$ where $|\mu_k - PC1_{i,j,1}| =$
      $\min_{l \in K} |\mu_l - PC1_{i,j,1}|$;
  **join** *no_of_processes*;
  recompute class means;
end

Algorithm SIMD_Kmeans(its, x, threshold, K, cluster, sigs)
Input: its (maximum number of iterations)
x (three-dimensional image)
threshold (iteration termination criteria)
K (number of clusters)
Output: cluster (two-dimensional cluster assignment array
for x)
sigs (cluster signatures)
begin
  call SIMD_Initialize_Means(x, K, m); (where
$m^{(1)}, \ldots, m^{(K)}$ are the means for clusters $1, \ldots, K$)
  for $it := 1$ step 1 until its do
    begin
      for $k := 1$ step 1 until $K$ do
        begin
          $sum^{(k)} := 0$;
          $n_k := 0$;
        end
      pixels_changed := 0; (initialize number of pixels
      that change assignment to zero)
      private i, j, min_distance, min_cluster, k, distance, p
      shared rows, cols, K, b, x, m, cluster
      for $i := 1$ step 1 until rows fork ASSIGNLOOP
      ASSIGNLOOP:
        for $j := 1$ step 1 until cols do
          begin
            min_distance := large_number; (initialize
            minimum distance squared)
            min_c := 0; (nearest cluster index)
            for $k := 1$ step 1 until $K$ do
              begin
                distance := $\sum_{p=1}^{b}(x_p^{(i,j)} - m_p^{(k)})^2$;
                if (distance < min_distance) then
                  begin
                    min_c := k;
                    min_distance := distance;
                  end
              end
            (add to local copies for later reduction)
            $sum^{(min\_c)} := sum^{(min\_c)} + x^{(i,j)}$;
            $n_{min\_c} := n_{min\_c} + 1$;
            if ($min\_c \neq cluster_{i,j}$) then
              begin
```

```
                    $cluster_{i,j} := min\_c$;
                    (add the following for later reduction)
                    $pixels\_changed := pixels\_changed + 1$;
                  **end**
              **end** ($j$ loop)
          reduce *sum, n, pixels_changed* across *no_of_processes*;
          **join** *no_of_processes*;
          **for** $k := 1$ **step** 1 **until** $K$ **do**
            **if** $(n_k \neq 0)$ **then**
              $m^{(k)} := sum^{(k)}/n_k$;
            **else**
              Delete Cluster $k$;
          **if** $(pixels\_changed/no\_of\_pixels < threshold)$ **then**
            **exit** loop;
        **end** ($it$ loop)
      **for** $i := 1$ **step** 1 **until** $rows$ **fork** OUTPUTLOOP
      OUTPUTLOOP:
        **for** $j := 1$ **step** 1 **until** $cols$ **do**
          update $sigs_k$ for cluster $k$ to which $x(i,j)$ belongs
    **end**
```

The final PIGSCR algorithm follows.

```
*Algorithm SIMD_PIGSCR(image, P, DR, IS,*
*IS+,$\alpha$, $p_0$, its, classes, kits, threshold, sigs,*
*DR_image, IS_image, IS+_image)*
**Input:** *image* (three-dimensional image)
$T$ (set of training data containing x,y coords and an
informational class value)
*DR, IS, IS+* (Boolean values corr. to output)
$\alpha$ (type-I error rate)
$p_0$ (homogeneity threshold)
*its* (number of iterations for main loop)
*classes* (number of classes to create in $K$-means loop)
*kits* (number of $K$-means iterations)
*threshold* (loop-ending criteria for $K$-means)
**Output:** *sigs* (set of pure signatures)
*DR_image* (image resulting from maximum
likelihood on input image)
*IS+_image* (image resulting from maximum
likelihood on impure pixels)
*IS_image* (image resulting from impure pixels
being recoded in unsupervised image)
**begin**
  *k_image := image* (*k_image* is input to $K$-means)
  **for** $i := 1$ **step** 1 **until** *its* **do**
    **begin**
```

```
        if (all pixels in $k\_image$ are zero) then
          exit loop;
        call SIMD_Kmeans(kits, k_image, threshold,
        classes, k_classimage, ksigs);
        reset spectral class counts;
        $pure\_classes := 0$; (number of pure classes)
        for $j := 1$ step 1 until $|T|$ do
          begin
            $c := k\_classimage(T_x^{(j)}, T_y^{(j)})$;
            if ($c \neq 0$) then
              increment the informational class count
              for class of $P^{(j)}$ and total count within
              spectral class $c$;
          end
        for $j = 1$ step 1 until $|ksigs|$ do
          begin
            if ($(total_j(1 - p_0)) \geq 5$) then (where $total_j$
            represents the count of pixels in sig $j$)
              begin
                Determine $N_{maj}$ and $C_{maj}$ where
                $N_{maj}$ is the no. of pixels in
                majority info class, $C_{maj}$
                $\hat{p} := \frac{N_{maj}}{total_j}$;
                $Z := \frac{(\hat{p} - p_0 - .5/total_j)}{\sqrt{p_0(1-p_0)/total_j}}$;
                if ($Z > Z(\alpha)$) then
                  begin
                    increment $pure\_classes$;
                    add $ksig(j)$ to group of $sigs$;
                    mask $j$ from $k\_image$ in parallel;
                    recode where $k\_classimage = j$ to
                    $C_{maj}$ in $class\_image$ in parallel;
                    (where $class\_image$ is the stacked
                    output of the $K$-means)
                  end
              end
          end
        if ($pure\_classes = 0$) then
          exit loop;
      end
    if ($DR$) then
      call SIMD_Maximum_Likelihood(sigs, image,
      DR_image);
    if ($IS+$) then
      begin
        call SIMD_Maximum_Likelihood(sigs, k_image,
            IS+_image);
            IS+_image := IS+_image
            + class_image;
        end
    if (IS) then
        begin
            recode k_image non-zeros to number of info
            classes + 1 in parallel, store result in k_image;
            IS_image := k_image + class_image;
        end
end
```

6.1 Implementation Details

A good parallel algorithm will not approach its potential parallel speedup without additional consideration of the constraints of both the implementation hardware and the chosen programming paradigm. Theoretically with the shared memory paradigm, all memory is not only accessible by each process, but also equidistant. In practice this is rarely the case as physical limitations prevent such implementations. Therefore the programmer must understand the program's memory access patterns and then use the appropriate language constructs to best utilize the memory access provided by the underlying architecture.

On the SGI Altix specifically, although there is one global memory address space for all processors to access, each processor has one part of that memory that is closest and fastest to access. The methodology for efficiently using this nonuniform memory access (NUMA) involves (local memory) programming tricks as well as tools that are separate from the program itself.

On the SGI Altix, there is a first touch principle within a program whereby the processor that first "touches" a data value will physically place that data in the memory closest to itself. For example, if after an array is allocated, one processor initializes that array, the physical location of the entire array will reside in the memory closest to that processor. If, however, each processor initializes some portion of the array, that portion will be closest to the processor that initializes it. In an OpenMP application on an SGI Altix architecture, it is important to initialize all shared variables in the same manner that each will be used, and it is important to access those variables consistently throughout the parallel application.

Outside of the application program, there are certain tools available to aid with the memory placement to facilitate good parallel performance. One of these tools specific to the SGI platform is `dplace`, a tool that binds a physical block of memory to a specific process. `Dplace` ensures that processes do not migrate from processor to processor, therefore negating any attempt within the program to keep processes close to the memory each accesses. `Dplace` can be a very useful tool when migrating processes occur during parallel runs, but all tests run on PIGSCR showed that `dplace` had virtually no effect on speed. In fact, using `dplace` would often result in slightly slower run times.

Chapter 7: PIGSCR EXPERIMENTAL RESULTS AND DISCUSSION

7.1 Data Description

PIGSCR was tested using three mosaicked Landsat Enhanced Thematic Mapper Plus (ETM+) satellite images taken from Landsat Worldwide Reference System (WRS) paths 15, 16, and 17, covering the majority of the state of Virginia, USA. These images, which will hereafter be referred to as VA15, VA16, and VA17, respectively, were obtained on April 28, 2004; May 8, 2005; and November 2, 2003, respectively. They are roughly similar in size, with VA15, VA16, VA17 being 1.1, 1.2, 0.9 gigabytes, respectively. The training data was created by the interpretation of point locations from a systematic, hexagonal grid (see Figure 1) over Virginia Base Mapping Program (VBMP) true color digital orthophotographs[1] [93]. The source photography data was acquired in the Spring of 2002 at a scale of 1:4,800, and after orthocorrection was resampled to a 1 meter spatial resolution. The Landsat data were used to perform a two-class classification, forest or non-forest. All three images are shown in Figures 2, 3, and 4, respectively. The map figures in this document were created using ESRI ArcGIS$^{(TM)}$ 9.2.

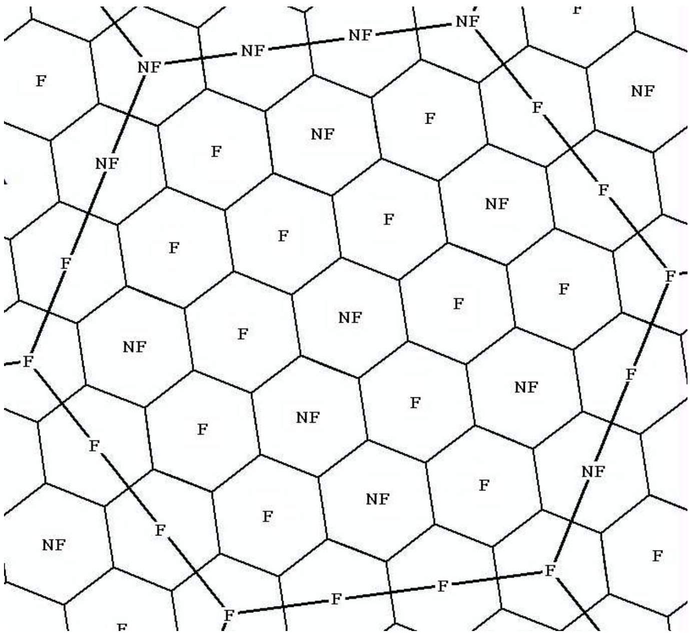


Figure 1. Hexagonal grid used for interpretation of training data.

For the purpose of accuracy assessment, a major component of the parameter selection process, validation data in the form of point locations at the center of USDA Forest Service Forest Inventory and Analysis (FIA) ground plots were used to validate the classifications [6][76]. Only homogeneous FIA plots were used (either 100 percent forest or non-forest), and these plots were obtained between 1997 and 2001. The lapse in years between the dates of the FIA plots and those of the classified Landsat imagery likely contributed to a lower classification accuracy in this area of rapid land cover change.

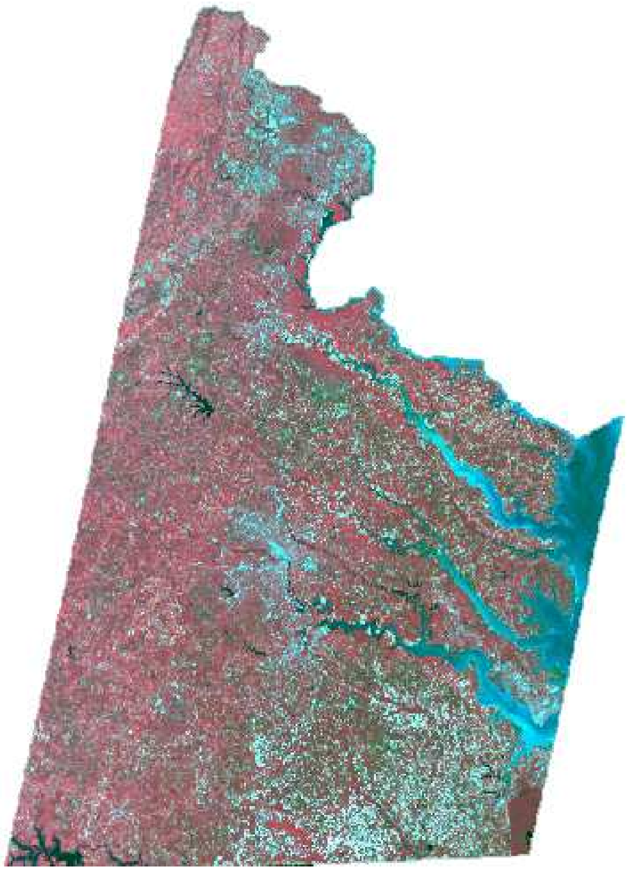


Figure 2. VA15 (Landsat ETM+ path 15).

7.2 Parameter Selection

One possible benefit of using a faster algorithm is the ability to make several runs to adjust input parameters to obtain the highest possible classification accuracy. Hartigan [41] mentions that because K-means is highly sensitive to input parameters and initialization of cluster means, several different initial numbers of clusters might be used to find the best clustering. Furthermore, the homogeneity threshold determining pure spectral classes in PIGSCR may be raised or lowered, potentially affecting overall classification accuracy.

Past applications of IGSCR have demonstrated success using homogeneity thresholds of 90 or 95 percent and 100 clustering classes [55][67]. Although it is reasonable to suspect those parameters might result in a good classification in this instance, there are differences between past applications of IGSCR and this application that merit additional consideration.

First, training was performed using areas of interest with spatially contiguous points in past applications. The training data used in this classification are evenly distributed points that are interpreted over a hexagonal grid placed across the Commonwealth of Virginia using high resolution 1:4,800 orthoimagery. This has resulted in fewer training points that might reduce the overall accuracy of the classification [83] but these points should be a more representative sample of the entire image. Secondly, this implementation of PIGSCR is potentially different from the previous implementation that relied on commercial closed source software libraries to perform classifications. There is no way to verify that the previous implementations of the decision rule and clustering algorithms are mathematically

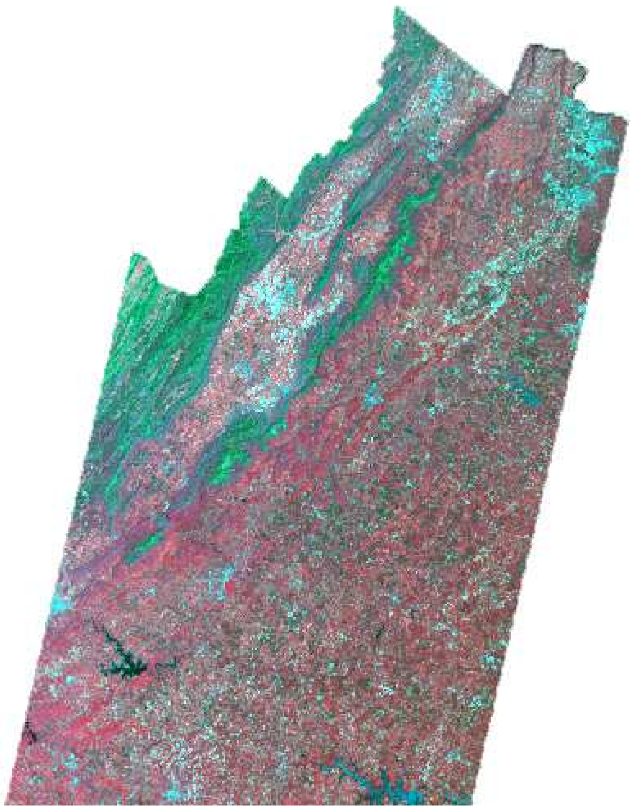


Figure 3. VA16 (Landsat ETM+ path 16).

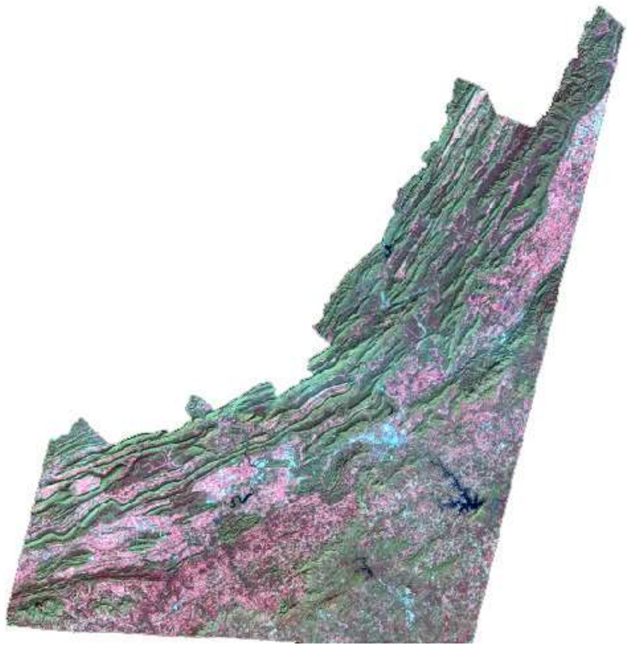


Figure 4. VA17 (Landsat ETM+ path 17.

equivalent to the open source implementations, and the description of the clustering algorithm previously used is unclear.

Finally, independent of changes in training methodology and algorithm implementation, it is likely, although not proven, that the ideal set of parameters is dependent on the data used in the classification. If this is indeed the case, there is value in reexamining ideal input parameters for each new pairing of training data and images.

In order to determine a homogeneity threshold and number of initial K-means clusters, several exhaustive combinations within a range were used for an application of PIGSCR on VA16 and resulting accuracies were compared as shown in Table 3. The homogeneity threshold varied between 70 and 95 percent in increments of 5 percent, and the initial number of classes varied between 40 and 100 in increments of 10. The accuracies ranged between 84.4 and 88.9 percent, with the highest, 88.9 percent, resulting from using 70 classes and 70 percent homogeneity. VA16 is representative in landcover and size of VA15 and VA17, and therefore 70 classes and 70 percent were used for the purpose of benchmarking PIGSCR using those images. The resulting classified images are shown in Figures 5, 6, and 7, and the classification accuracies for VA15 and VA17 are 89.5 percent and 90.5 percent, respectively.

Table 3. Factorial Analysis for VA16.

	Number of Classes						
p_0	40	50	60	70	80	90	100
95	86.7	86.6	86.5	84.8	86.3	85.4	84.4
90	86.8	87.0	84.4	88.0	85.8	85.8	87.3
85	87.4	88.6	88.1	87.4	87.5	88.5	88.0
80	88.3	87.8	88.2	87.7	87.6	87.8	88.1
75	87.5	87.9	87.0	87.4	87.4	87.8	87.9
70	87.0	88.2	88.2	**88.9**	88.5	87.7	88.4

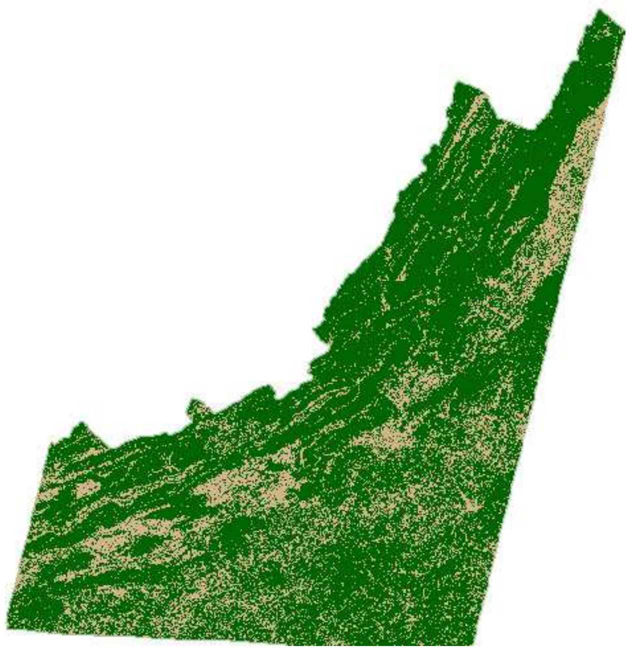


Figure 5. Classification of VA17 (Landsat ETM+ path 17), green is forest and tan is non-forest.

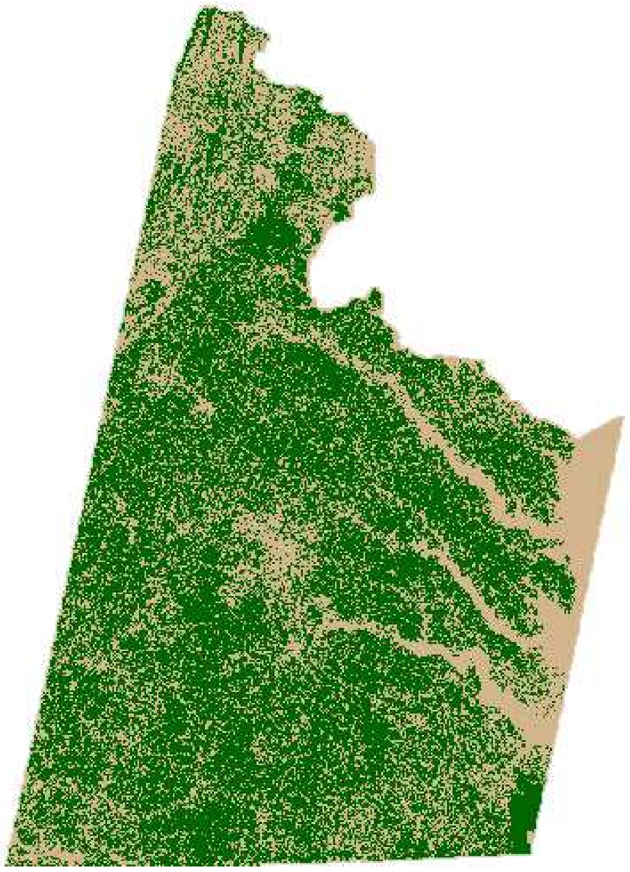


Figure 6. Classification of VA15 (Landsat ETM+ path 15), green is forest and tan is non-forest.

7.3 Parallel Results

PIGSCR was tested with the previously described images using one, two, four, eight, and twelve processors of an SGI Altix 3300 with 24 GB of RAM and twelve 1.3 GHz processors and using 16, 32, and 64 processors of an SGI Altix 3700 with 64 1.6 GHz processors and 256 GB of RAM. Figures 8 and 9 show speedup as a function of number of processors when compared with the execution time for one processor and two processors, respectively. The solid black line shows ideal speedup, i.e., speedup equal to the number of processors used. The dashed line represents VA15, the dash-dot line, VA16, and the dotted line, VA17. The lines in Figure 8 are superlinear, which do not indicate much about the effect of increased processing speed on overall execution time. The drastic decrease in processing time is likely the result of additional memory available with additional processors. The lines in Figure 9 take the lack of memory available to one processor into account and therefore most probably best demonstrate parallel speedup. As the number of processors is increased, the speedup of PIGSCR increases with each of the three test images, as shown in both Figure 8 and 9, but the rate of speedup is decreasing. Table 4 is included to show how the two speedups (based on one and two processors) relate to actual execution times for the three images. Speedup is a good indication of how well the parallel program is using additional computing resources, but decreased execution time is what is important to a user.

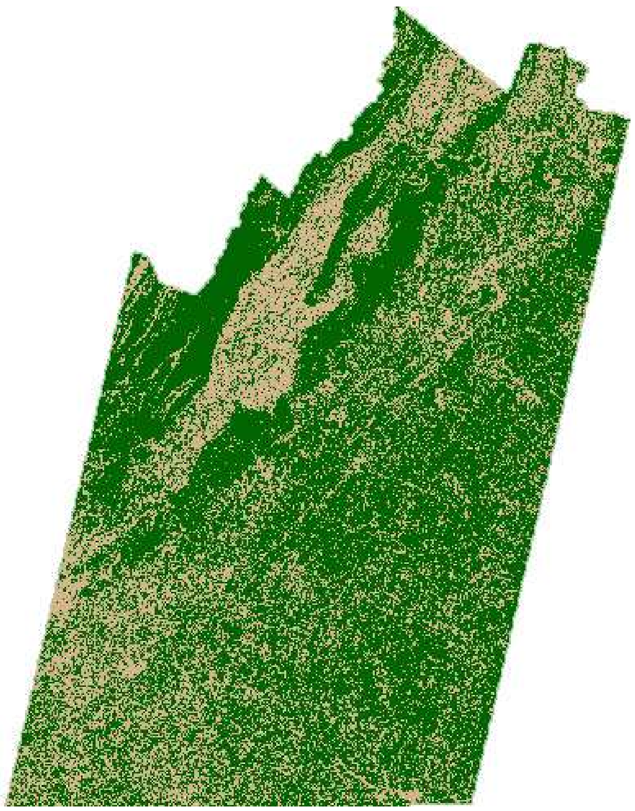


Figure 7. Classification of VA16 (Landsat ETM+ path 16), green is forest and tan is non-forest.

All three images are similar in size, so similar processing times and speedups might be expected. Variations in execution time and speedup can be accounted for by different numbers of iterations within PIGSCR, depending on the purity of the clusters that are created by unsupervised classification. VA17 showed the least speedup, as would be expected considering VA17 is the smallest of the three images. Furthermore, VA17 required fifteen iterations of PIGSCR while VA16 required nine and VA15 required six. As more iterations are completed, the parallel speedup slows as K-means is run on successively smaller images. This is because the time spent executing parallel sections of code decreases while the time spent executing serial sections remains constant. This also explains why VA15 with only six iterations had better parallel speedup than VA16 (a larger image) with nine iterations.

Table 4. Execution Time(sec) for VA15, VA16, and VA17, $p =$ number of processors.

p	VA15	VA16	VA17
1	16,076	27,229	30,311
2	4,056	5,997	4,353
4	1,809	2,949	2,268
8	852	1,451	1,194
12	576	1,050	888
16	426	763	708
32	283	494	495
64	209	355	426

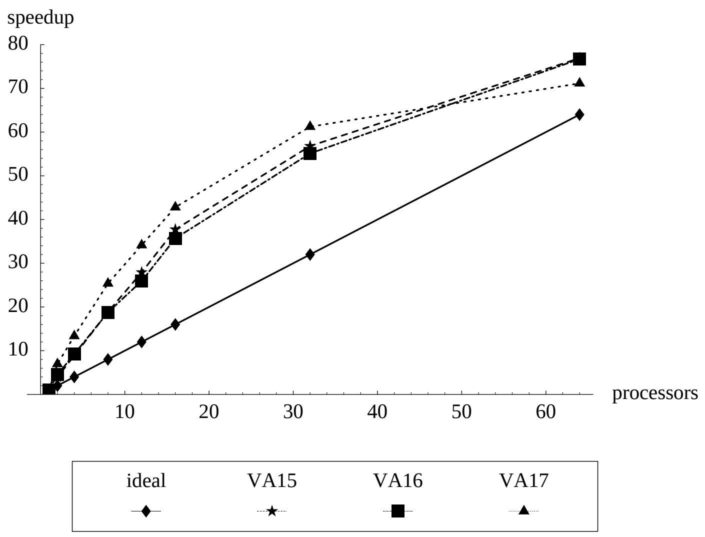


Figure 8. Speedup with respect to one processor.

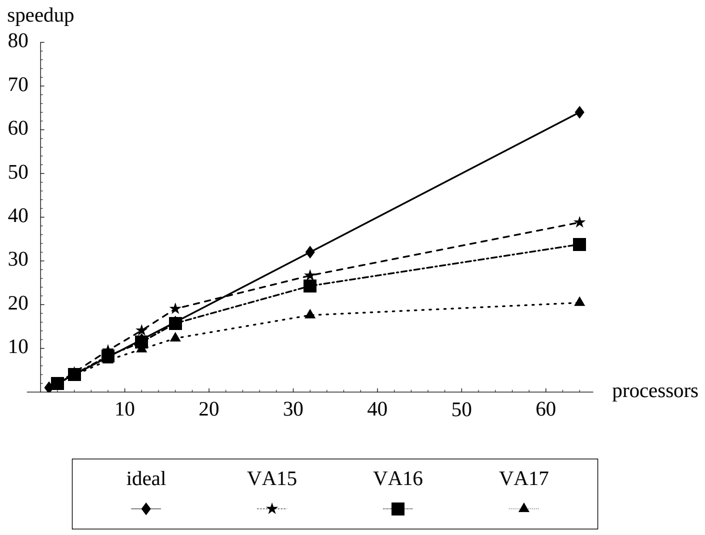


Figure 9. Speedup with respect to two processors.

7.4 Discussion

Theoretically, an application that parallelizes perfectly would have a parallel speedup equal to the number of processors used, as shown by the solid lines in the figures. In reality, there are a number of factors that influence parallel speedup, including constraints of the underlying architecture, parallel communication and management overhead, and inherent serial portions of the algorithm that is written in parallel. The above speedup graphs provide evidence that many of these different factors influence the parallel speedup of PIGSCR.

Considering only processor number and speed, it would seem impossible for the actual realized speedup to be greater than ideal speedup, as is the case in Figure 8 and 9. In order to explain greater than ideal speedup, or superlinear speedup, computer memory must be considered. Different types of memory are accessed at different speeds. Cache memory is the fastest, followed by random access memory (RAM), and the slowest form of memory (virtual) is the disk drive. If there is more data in an application than can fit in one fast type of memory, then the processor must rely on the slower access of another type of memory in order to process all of the data. On the SGI Altix platform specifically, each processor has individual cache memory, and the superlinear speedup observed can be explained by the addition of more cache with the addition of more processors. Furthermore, although adding more processors does not technically add more RAM, because of the Altix's NUMA architecture, adding more processors will decrease the distance (access time) between each processor and the RAM it accesses, if the application is programmed to take advantage of the architecture as described above.

Except for the most trivial parallel applications, parallel codes will not realize perfect speedup because there is some part of the algorithm that is inherently sequential (Amdahl's Law, [1]). There are several sections of PIGSCR that are still executed serially in the present implementation. Furthermore, because PIGSCR relies on data parallelism, the portion of the computation that is run in parallel depends on the size of the dataset that is processed. As the size of the dataset decreases, the serial section of the algorithm consumes a greater percentage of overall processing time compared with the parallel sections. With each iteration of PIGSCR, a progressively smaller dataset is clustered in parallel with K-means, meaning that the part of K-means that can be run in parallel decreases in size while the amount of serial processing remains the same. This explains the leveling off of the speedup past sixteen processors.

A leveling off of speedup (or decrease in speedup in some cases) may also occur as a result of the underlying architecture's mechanism for communication. In order for each processor to access the global memory space, it must send and receive data using the (hardware) global communication infrastructure. When a processor accesses the memory block that is physically closest on the NUMA architecture, there is less competition for shared communication resources. However, when processors are competing to access the same memory, for example some of the shared variables such as the cluster or class information, slower memory access times could occur. The computation slows down any time two or more processors attempt to access the same memory, and the potential for this to occur is greater as more processors are used. This increased memory contention coupled with a decreasing work load per processor per PIGSCR iteration, results in less than ideal speedup for a large number of processors.

Chapter 8: PIGSCR CONCLUSIONS

Prior to this work, IGSCR classification was performed using a “black box” proprietary software library, and classification of a full Landsat scene required several hours of processing time. With the creation of a portable and open source serial version of IGSCR, the black box was removed, allowing analysts to verify and modify the algorithms that are used. The large speedup observed was the result of adding both memory and processors, and therefore does not necessarily indicate good parallel speedup in a technical parallel computing sense; however, these results indicate real time that is saved for a real user. The specifications of one processor used are very similar to a very powerful standalone desktop computer. These results show that execution times can be reduced from several hours to several minutes by using only a modest number of processors in a parallel environment. Hence with parallel computing it is feasible to perform many runs of IGSCR in order to obtain the ideal combination of input parameters leading to the best classification accuracy.

PIGSCR showed good parallel speedup through 16 processors, however, speedup showed signs of leveling off at 32 and 64 processors when compared to the execution time on two processors. Further work on a PIGSCR algorithm needs to be done in order to justify using a larger (distributed memory) supercomputer to run PIGSCR. Clustering algorithms and decision rules that are more amenable to a parallel environment might be considered in a distributed memory implementation. Furthermore, adding more underlying classification algorithm options available to a user is now possible. The previous implementation of IGSCR that used proprietary classification libraries allowed less flexibility, limiting a user to specific classification algorithms. Additionally, a more detailed study of the IGSCR framework is possible with this increased flexibility.

Section II. SVD-BASED FEATURE REDUCTION.

Chapter 9: SVD-BASED FEATURE REDUCTION INTRODUCTION

In the remote sensing and image processing discipline, large data volumes and slow processor speeds have necessitated feature reduction. Since the cost of classifications is dependent upon the number of discriminating features (bands) associated with each pixel in multispectral space, it is desirable to reduce the number of features in a dataset [77]. Even as processing speeds increase due to faster computers and better algorithms, such as the parallel iterative guided spectral class rejection (PIGSCR) classification algorithm [71], the need for feature reduction methods remains as modern sensors increase in sensitivity.

This thesis also presents a feature reduction method using the singular value decomposition (SVD). The SVD emerges as an ideal candidate for feature reduction of remotely sensed images due to inherent collinearity of the brightness value vectors in geographic data. The SVD is applied in a new way, drastically reducing the computer processing time and memory requirements and therefore making the SVD feasible for feature reduction in large datasets. This work examines various feature reduction methods for the PIGSCR classification algorithm, and demonstrates that the proposed SVD method significantly decreases classification execution times while not negatively affecting classification accuracy, and the SVD outperforms some other commonly used feature reduction methods such as principal components analysis. The SVD has potential for improving classification accuracies as too many features for a given training set can reduce classification accuracy [45], and the SVD can enable the removal of noise (corresponding to small singular values) similar to the Fourier Transform.

Chapter 10: FEATURE REDUCTION BACKGROUND

Traditional methods of feature selection include the use of separability indices such as divergence, the Jeffries Matusita Distance, and transformed divergence [77]. A set of spectral classes are analyzed to determine which combinations of bands will result in the greatest separability (greatest distinction between classes), and only those bands are used for the ensuing classification. Another popular approach is that of feature reduction, where the image is transformed to a new coordinate system requiring (hopefully) fewer bands to accurately represent the image. Most feature reduction methods do not require analysis of training data, making this an attractive option for a classification method that does not require training, such as unsupervised classification methods and some hybrid classification methods. Standard feature reduction methods include principal components analysis (PCA) (also called Karhunen-Loeve analysis) [48], maximum noise fraction (MNF, also called minimum noise fraction) [38], canonical analysis [31], and the Kauth-Thomas tasseled cap transformation [54]. The tasseled cap transformation has a fixed axis and is therefore somewhat confined in its application. The other feature reduction methods mentioned are transformations that align the data along axes of decreasing variance, and the resulting low order de-correlated bands are sufficient to perform a classification in many cases. However, Lowitz [65] and Chang [23] have shown that sometimes the higher order components resulting from such transformations are necessary to differentiate between classes in a classification. Also, the axes generated using PCA may not allow for accurate classification of the data using fewer dimensions while a different set of axes exists that will allow for class discrimination using fewer dimensions, the premise upon which canonical analysis is based [77]. Furthermore, rather than directly revealing the rank and basis of the data from the data itself, these methods attempt to reveal these attributes indirectly from a summary of the data, such as the covariance matrix. This explains why the resulting PCA data has full rank (no reduction is possible), but it is still possible that a different alignment of the data will result in a feature reduction.

A mathematical construct that directly reveals the rank and corresponding ideal basis of a dataset is the singular value decomposition (SVD). For a dataset in n-dimensional space, for any $k < n$, the SVD will show the ideal basis for representing that data using only k dimensions, as will be explained in a later chapter. If the SVD reveals that the dataset is full rank and no feature reduction is possible along the calculated axes, then no axes exist for which a reduction is possible. The SVD reveals the ideal subspace for the data based on the entire dataset, while axes transformations such as PCA attempt to do so with a limited summary of the data (i.e., the covariance matrix). k-dimensional subspaces (for any specific k) revealed by other constructs will never be mathematically closer to the original subspace than the k-dimensional subspace revealed by the SVD.

In remote sensing applications, the SVD is a popular alternative factorization to QR factorization for solving least squares problems [66][25]. The use of the SVD as a feature reduction tool has been limited in remote sensing as the storage and processing are expensive, especially for large datasets such as entire images [28][42]. In the discipline of chemistry, van den Broek et al. [92] used the SVD to reveal the rank and reduce the data dimension of multivariate images.

Chapter 11: FEATURE REDUCTION WITHIN PIGSCR

This thesis has shown that PIGSCR is significantly faster than IGSCR, and therefore input parameters may be selected using a factorial analysis of input parameters to determine the specific combination of parameters that results in the highest classification accuracy. This approach has been extended to factorially determine which combination of bands would produce the best classification accuracy for a specific set of input parameters, dataset, and training dataset. Initial experimental runs on images containing six bands demonstrated that the most accurate classifications on all experimental images typically used all six bands. This approach was then further extended to experimentally determine the most accurate combination of bands for each unsupervised classification performed within each iteration of IGSCR. The resulting PIGSCR algorithm is presented below.

Algorithm PIGSCR with feature reduction

```
User inputs an image, training dataset, validation dataset, maximum number of iterations,
$\alpha$ and $p_0$ for the homogeneity test,
and number of clusters to be created in each iteration.
The output that is produced includes a set of pure class signatures
that is used in the final supervised classification, and any or all of
the following three classification images: DR, IS, IS+
begin
  do until maximum number of iterations is reached,
  no pixels remain in original image,
  or no pure classes are found:
    do for each possible band combination:
      Cluster remaining pixels using parallel clustering algorithm.
      do for each point in the training data set, in parallel:
        determine the spectral class assignment based on the clustering,
        and increment the appropriate informational class count
        and spectral class count
      end do
      do for each cluster, in parallel:
        use the homogeneity test given above to determine informational class assignment.
      end do
      do in parallel:
        Recode all pixels in the unsupervised classification to informational classes
      end do
      Create accuracy assessment matrix and determine overall classification accuracy
      for band combination
    end do
    Keep unsupervised classification image for most accurate clustering
    do for each cluster, in parallel:
      if a particular cluster is pure, then
        Add that cluster's signature to the set of pure class signatures
        and mask all vectors/pixels belonging to that class out of the input image.
        Recode all pixels belonging to that spectral class in the
        unsupervised classification image to the value of the assigned informational class.
      end if
    **end do**
  **end do**
  Perform a parallel supervised classification
  on the input image using the pure signature set.
  **do** in parallel:
    Recode all impure classes in the unsupervised classification image
    to a value reserved for impure classes and add this to the
    unsupervised classification image for a second output image.
  **end do**
  Perform a parallel supervised classification on only the impure pixels and
  add them to the unsupervised classification image to produce a third
  output image.
**end**
```

This feature reduction algorithm for PIGSCR produced marginally better accuracies in some classifications performed on images with six bands. Unfortunately, each iteration requires that $2^{bands} - 1$ classifications be performed to determine the most accurate combination of bands, and it is therefore not feasible for images with large numbers of bands. The six band classifications required that 63 individual unsupervised classifications be run per iteration, and an image with as few as ten bands would require over 1000 classifications be performed for each iteration of IGSCR. Additional algorithmic improvements and additional processing power would be required to make this band selection algorithm a feasible option.

Chapter 12: SVD-BASED FEATURE REDUCTION

At a high level, the SVD reveals the minimum number of dimensions required to represent a matrix or linear transformation [61] [89]. Often multi-dimensional data may be represented equivalently (or approximately so) in fewer dimensions due to redundancies in data. If a set of n-dimensional vectors all lie in a k-dimensional subspace, $k < n$, then each n-vector effectively has only k degrees of freedom, and can be uniquely described by k numbers. The natural correlations that occur in nature make the SVD a good candidate for feature reduction. Furthermore, if no reduction is possible, this will be shown by the magnitudes of the singular values revealed by the SVD.

The SVD of a linear transformation $A : R^n \longmapsto R^m$ is

$$A = U\Sigma V^t,$$

where $U : R^m \longmapsto R^m$ is orthogonal ($U^tU = I$), $\Sigma : R^n \longmapsto R^m$ is a diagonal matrix whose diagonal elements are called the singular values of A, and $V^t : R^n \longmapsto R^n$ is orthogonal as well. Each linear transformation in the SVD is expanded and shown as follows, assuming $m < n$:

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$
$$= \begin{pmatrix} u_{11} & \dots & u_{1m} \\ \vdots & \ddots & \vdots \\ u_{m1} & \dots & u_{mm} \end{pmatrix} \begin{pmatrix} \sigma_{11} & \sigma_{12} & \dots & \sigma_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{m1} & \sigma_{m2} & \dots & \sigma_{mn} \end{pmatrix} \begin{pmatrix} v_{11} & v_{12} & \dots & v_{1n} \\ v_{21} & v_{22} & \dots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n1} & v_{n2} & \dots & v_{nn} \end{pmatrix}.$$

Since entries σ_{ij} when $j > m$ are all zero, the product ΣV^T will produce entries of zero for rows $m + 1$ through n. The SVD may be rewritten as

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$
$$= \begin{pmatrix} u_{11} & \dots & u_{1m} \\ \vdots & \ddots & \vdots \\ u_{m1} & \dots & u_{mm} \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_m \end{pmatrix} \begin{pmatrix} v_{11} & v_{12} & \dots & v_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{m1} & v_{m2} & \dots & v_{mn} \end{pmatrix}.$$

This shows that a column $A_{\cdot i}$ of A, an m vector, can be expressed as a linear combination of the m basis vectors in U ($U_{\cdot 1}, U_{\cdot 2}, \dots, U_{\cdot m}$), using the singular values in Σ ($\sigma_1, \sigma_2, \dots, \sigma_m$), and the ith column $V^t_{\cdot i}$ in V^t. The diagonal elements of Σ are nonnegative, and can be ordered such that $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_m$. If some entries on the diagonal of Σ are zero, then for some k, $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k > \sigma_{k+1} = \dots = \sigma_m = 0$. Using the above logic that allowed a reduction in the number of rows in V^t from n to m, the number of columns in U can be reduced to k, the number of rows and columns of Σ can be reduced to k, and the number of rows in V^t can be reduced to k, yielding

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$
$$= \begin{pmatrix} u_{11} & \dots & u_{1k} \\ \vdots & \ddots & \vdots \\ u_{m1} & \dots & u_{mk} \end{pmatrix} \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \sigma_k \end{pmatrix} \begin{pmatrix} v_{11} & v_{12} & \dots & v_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{k1} & v_{k2} & \dots & v_{kn} \end{pmatrix}.$$

An operation such as a classification that would be performed on the entire $m \times n$ matrix A can now be equivalently performed on the entire $k \times n$ matrix ΣV^t where $k < m$, resulting in a reduction in the number of bands present in each vector. For practical purposes, singular values may in fact be nonzero yet be sufficiently close to zero to reduce the dimension of the data. The singular values $\sigma_{k+1}, \dots, \sigma_m$ represent distances from the subspace spanned by $U_{\cdot 1}, \dots, U_{\cdot k}$, and very small distances may not affect the operation that will be performed on the reduced data, such as classification. If none of the singular values on the diagonal are close to zero, then the data is already represented using as few dimensions as possible. However, this seems unlikely in the application of remote sensing as geographic data is naturally redundant.

Practically speaking, it would be necessary to think of a three-dimensional (pixel row, pixel column, data bands) image in two dimensions in order to take advantage of the feature reduction made possible by using the SVD. This would also be an expensive computation as the leading dimension of the matrix would be the number of bands, but the second dimension would be equal to the number of pixels in the image. One of the features of an SVD is that it reveals the basis vectors U that can be used to transform any vector from the original vector space (range of A) to the new vector space (range of $U\Sigma$). If a training dataset (columns of T) is truly representative of a particular image (columns of A), then the resulting SVD $T = \tilde{U}\tilde{\Sigma}\tilde{V}^t$ will also reveal a set of basis vectors for the range of A. Using this result, it is possible to perform the SVD on a training data matrix that is $m \times p$ in dimension, where p is the number of points in the training data set, and use the resulting SVD to transform A. In order to do this, it is necessary to project the columns of the matrix A onto the subspace spanned by the first k columns of $\tilde{U}$. This is accomplished by simply computing $(\tilde{U}_{\cdot 1}, \tilde{U}_{\cdot 2}, \dots, \tilde{U}_{\cdot k})^t A$. This result can be arrived at algebraically. Assume that the $m \times p$ matrix T is a submatrix of the $m \times n$ matrix A, and that dim range $A =$ dim range $T = k$. Then range $(U_{\cdot 1}, \dots, U_{\cdot k}) =$ range $A =$ range $T =$ range $(\tilde{U}_{\cdot 1}, \dots, \tilde{U}_{\cdot k})$, which means each column of A can be reduced to its k coordinates with respect to the orthogonal basis $\tilde{U}_{\cdot 1}, \dots, \tilde{U}_{\cdot k}$ of range $(\tilde{U}_{\cdot 1} \dots, \tilde{U}_{\cdot k})$. These coordinates are given by multiplying by $(\tilde{U}_{\cdot 1}, \dots, \tilde{U}_{\cdot k})^t$. After the original image has been transformed and the number of bands has been reduced, the PIGSCR classification algorithm is run with no modifications.

Chapter 13: DATA DESCRIPTION AND PREPARATION

PIGSCR with the SVD or PCA was tested using Landsat Thematic Mapper (TM) and Enhanced Mapper (TM/ETM+) images (path 17/row 34) acquired on December 1, 1999, March 6, 2000, June 10, 2000. The images from 1999 and 2000 were layered together to form a composite multi seasonal image, and all images were registered to a rectified 2003 image with an RMSE of 1/5 a pixel or less. Using Leica Geosystems Erdas Imagine^{TM} 8.7 software, registration was performed with 24 control points for each image pair, eight of which were randomly selected as check points. A first order transformation was used, and resampling was performed using nearest neighbor. Each of the three images used for the composite image was converted to reflectance in accordance with the Landsat 7 user's guide [60], and dark object subtraction was performed in ITT ENVI^{TM} 4.2 using the band minimum method. The composite image, which will be referred to as VA17C, contained 18 total bands. Figures 10, 11, and 12 show representative subsets of the VA17C image (bands 4, 3, and 2), the VA17C PCA (first three bands) and VA17C SVD (first three bands), respectively. Figure 13 shows the locations of the subset and VA17C in relation to each other and Virginia.

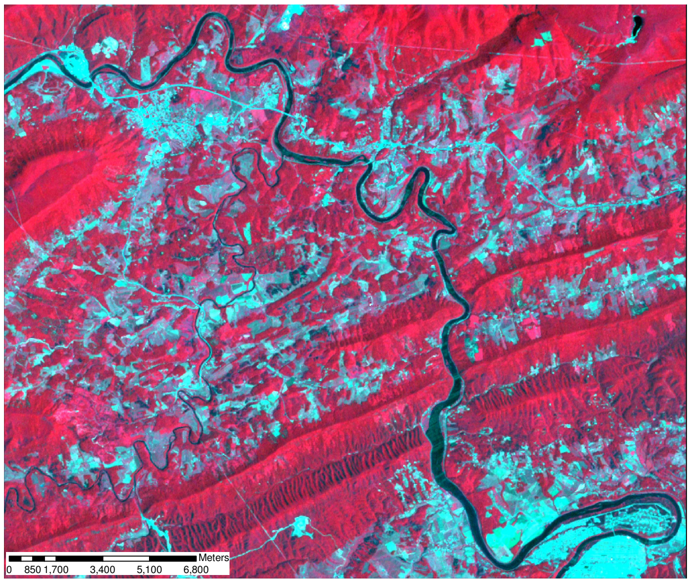


Figure 10. VA17C (Landsat TM/ETM+ path 17/row 34, bands 4, 3, and 2 shown) zoomed to a subset of interest.

The training data for these images was created by the interpretation of point locations from a systematic, hexagonal grid over Virginia Base Mapping Program (VBMP) true color digital orthophotographs [93]. The data were collected in the Spring of 2002 at a scale of 1:4,800, and were subsequently resampled to a 1 meter spatial resolution. The data were used to perform a two-class classification, forest or non-forest. For the purpose of accuracy assessment, validation data in the form of point locations at the center of USDA Forest Service Forest Inventory and Analysis (FIA) ground plots [76][6] were used to validate the classifications. Only homogeneous FIA plots were used (either 100 percent forest or non-forest), and these plots were visited between 1997 and 2001.

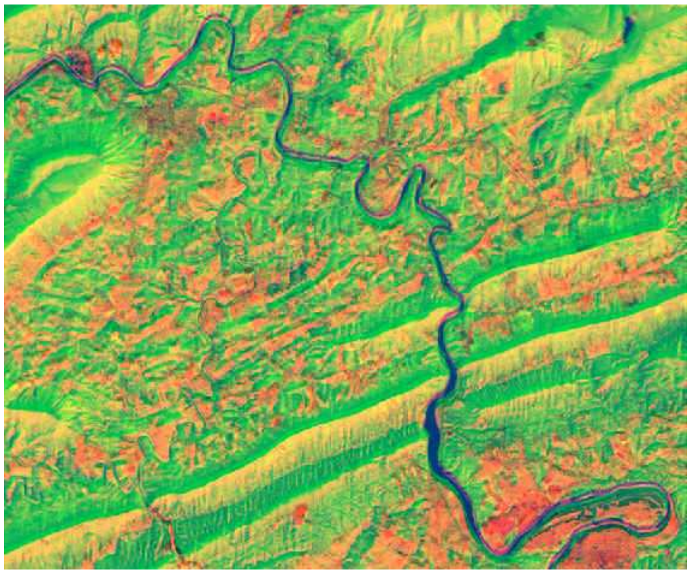


Figure 11. VA17C PCA (Landsat TM/ETM+ path 17/row 34, bands 1, 2, and 3 shown) zoomed to a subset of interest.

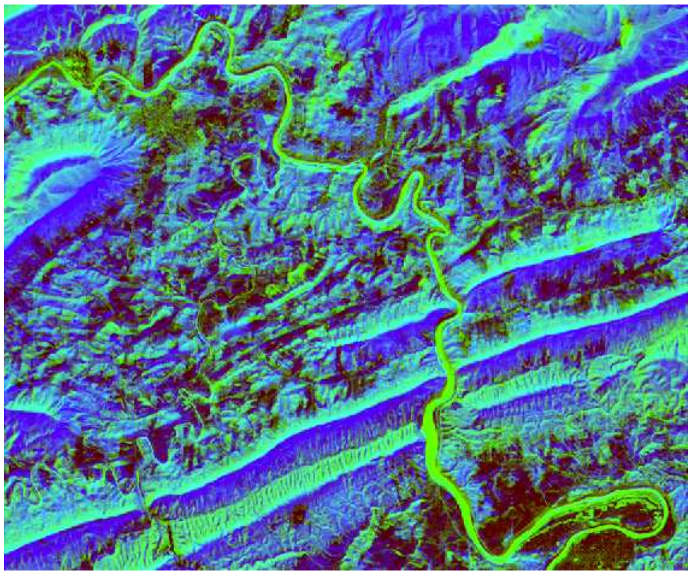


Figure 12. VA17C SVD (Landsat TM/ETM+ path 17/row 34, bands 1, 2, and 3 shown) zoomed to a subset of interest

A second dataset consisting of a multitemporal series of TM and ETM+ images from path 232/row 67 (1995-2002) acquired over Rondônia were used, each image registered to the 2001 image. The 2001 image was rectified by the US National Center for Earth Resources Observation & Science, and the remaining registrations were performed using Leica Geosystems Erdas Imagine^{TM} 8.5 with at least 50 control points and 25 randomly selected check points. The root mean squared error was less than 1/3 of a pixel for each image registration, and nearest neighbor resampling was used [51][96]. This multitemporal image will be referred to as AM232. Figures 14, 15, and 16 show representative subsets of

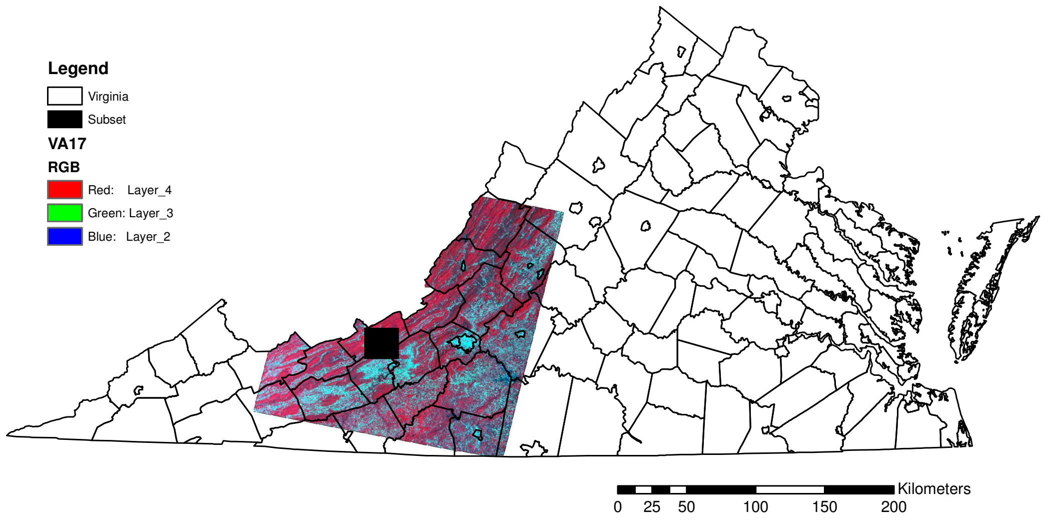


Figure 13. Locations of Virginia, VA17C (Landsat TM/ETM+ path 17/row 34), and zoomed subset shown in Figures 10—12 in relation to each other.

AM232 (bands 4, 3, and 2 from the 1998 image, corresponding to training data used for classification), AM232 PCA (first three bands), and AM232 SVD (first three bands). Figure 17 shows the locations of the subset and AM232 in relation to each other and Rondônia, Brazil.

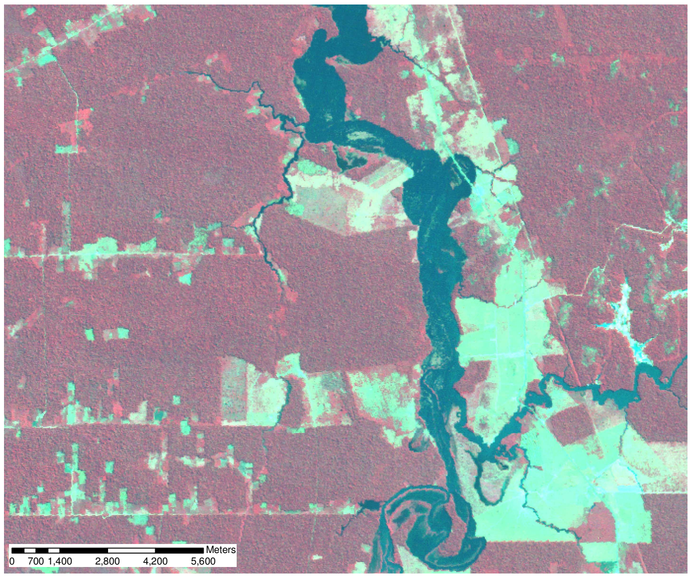


Figure 14. AM232 (Landsat TM path 232/row 67, bands 4, 3, and 2 of image acquired in 1998 shown) zoomed to a subset of interest.

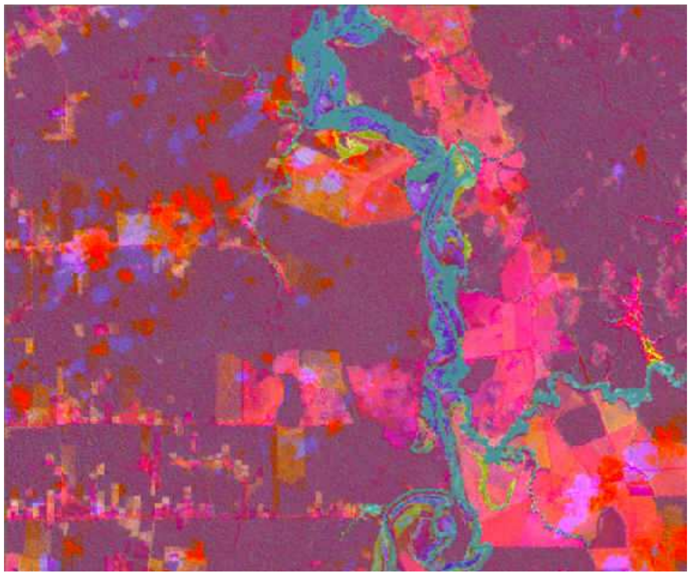


Figure 15. AM232 PCA (Landsat TM path 232/row 67, bands 1, 2, and 3 shown) zoomed to a subset of interest.

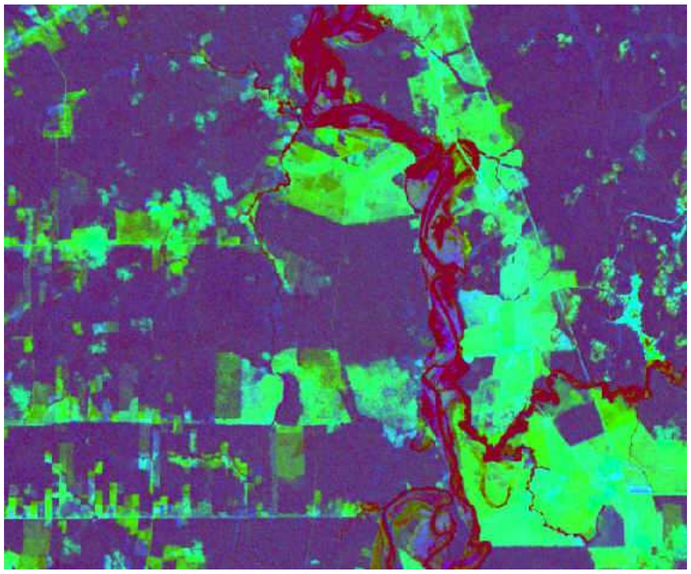


Figure 16. AM232 SVD (Landsat TM path 232/row 67, bands 1, 2, and 3 shown) zoomed to a subset of interest.

Collecting training data for this region was a challenge, as described by [51][96]. The training data used for a forest/non-forest PIGSCR classification was collected using detailed interviews, Landsat imagery, and detailed maps of various farms shown in the images. At least 67 pixels for each of the two classes (forest and non-forest) were present, and these points were used as seed pixels for a region growing algorithm (Erdas Imagine ^{TM} 8.5). Because of the difficulties in acquiring training data, a total of 200 validation pixels were randomly selected from the training data. This ensured a representation of edge and mixed pixels, which is necessary in order to avoid inflated classification accuracies [73].

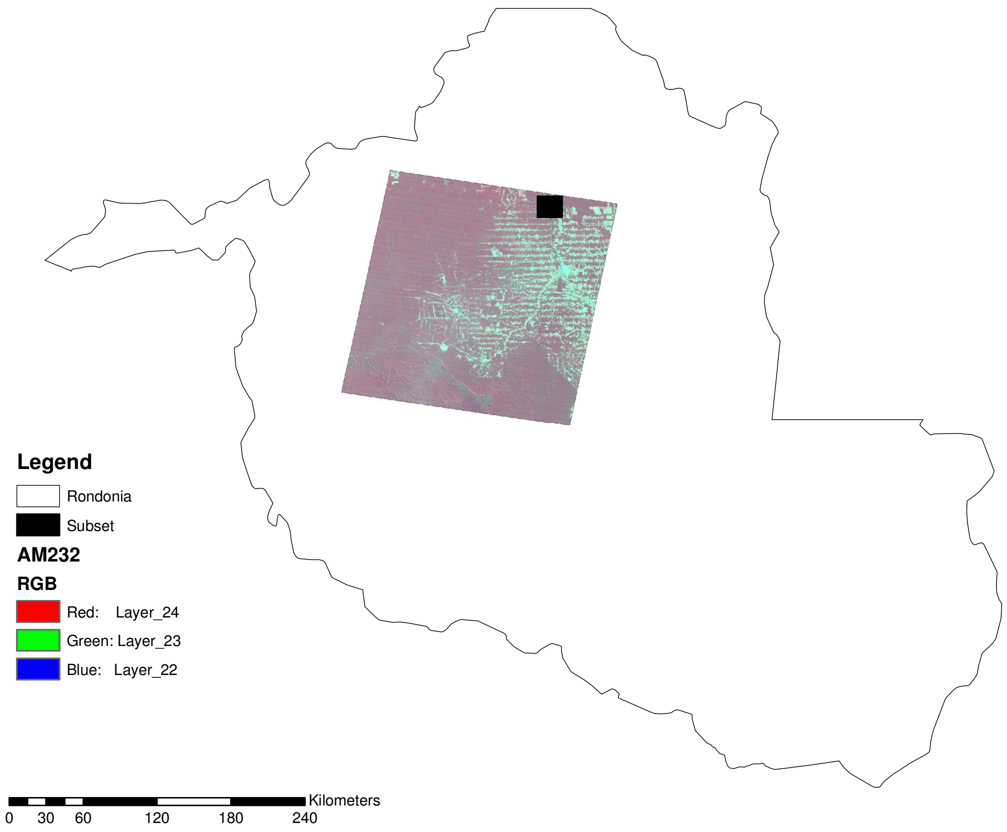


Figure 17. Locations of Rondônia, AM232 (Landsat TM path 232/row 67), and zoomed subset shown in Figures 14—16 in relation to each other.

Chapter 14: SVD RESULTS AND DISCUSSION

A two class classification (forest, non-forest) using PIGSCR with 80 initial classes and a homogeneity threshold of 80 percent was run on VA17C and AM232 and both of the above feature reduction methods were used. PIGSCR was applied to the composite VA17C image using all 18 bands, but applying PIGSCR to AM232 without feature reduction was infeasible considering the number of bands (52) in AM232. The above algorithms were implemented using Fortran 95 [33] and LAPACK [2]. These classifications were run using an SGI Altix 3300 with 24GB of RAM and twelve 1.3 GHz Itanium processors.

PIGSCR was applied to the VA17C SVD image using as few as four and as many as fourteen bands. Each resulting SVD classification was compared to the corresponding classification of the entire eighteen band VA17C image using McNemar's test for statistical significance of the difference between the two classifications as described by Foody [32]. A chi-square distribution with one degree of freedom and $\alpha = .05$ (3.841) was used where

$$\chi^2 = \frac{(x_1 - x_2)^2}{x_1 + x_2},$$

with x_1 being the number of pixels correctly classified by the first classification but incorrectly classified by the second, and vice versa for x_2. Using this test, all classifications using five or more bands from the SVD image were not statistically different from the classifications using all eighteen bands of VA17C. Out of eleven tests run (four through fourteen bands) four tests using the SVD of VA17C were actually marginally higher in accuracy than the classification accuracy of 88.62% using all eighteen bands. Figures 18 and 19 show the resulting classification images of VA17C and VA17C SVD (using six bands), respectively, where green pixels were determined to be forest and tan pixels were determined to be non-forest. Although these two classifications are statistically the same classification (as determined by McNemar's test), it is clear that using the SVD image resulted in the correct classification of the large river that is prominent in this scene, while the classification using all eighteen bands of VA17C determined most of the river to be forest.

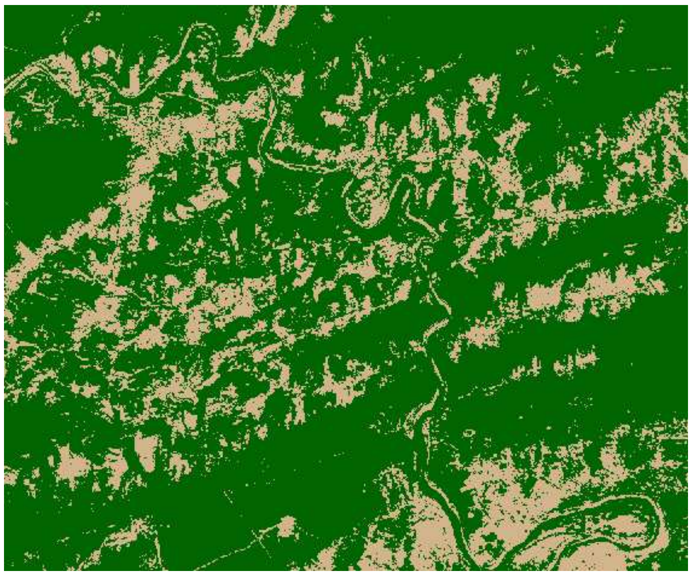


Figure 18. VA17C DR (Landsat TM/ETM+ path 17/row 34) classification, green = forest, tan = non-forest.

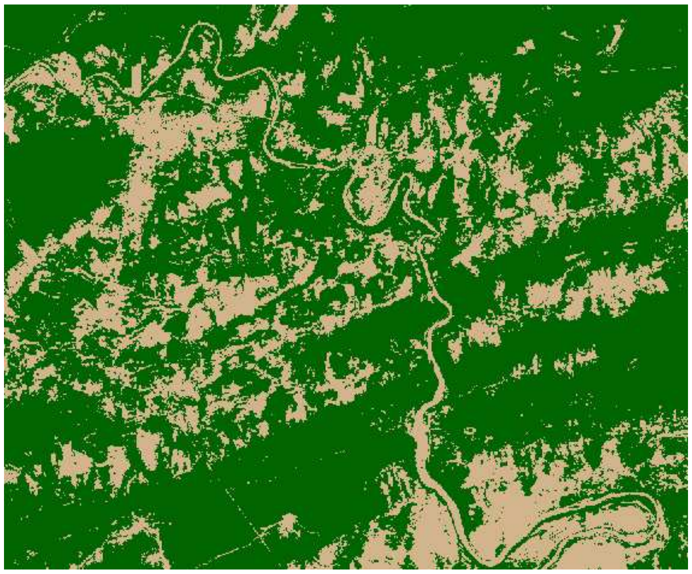


Figure 19. VA17C SVD DR (Landsat TM/ETM+ path 17/row 34) classification, green = forest, tan = non-forest.

Using all eighteen bands resulted in a PIGSCR execution time of roughly 1650 seconds, more than five times as long as the quickest run of just over 300 seconds using five bands of the SVD image. In general, using more bands increases the execution time, however, execution time is also dependent on the number of iterations performed during the iterative clustering and the number of pure classes used for the decision rule. It is therefore advantageous to use as few bands as possible when execution time is a great concern, and the accuracies resulting from using five through fourteen bands of the SVD image are consistently around 88%, both above and below the classification accuracy of VA17C. These consistent accuracies indicate that the choice of k, the singular value cutoff, is not crucial past five bands.

14.1 SVD versus PCA

Table 5. Comparison of Averages for PCA and SVD PIGSCR Classifications.

	SVD	PCA
VA DR accuracy	**88.18**	88.01
VA IS+ accuracy	**88.34**	88.16
VA execution time (secs.)	817	818
VA iterations	6.5	6.27
VA classes	78.2	77
AM DR accuracy	**92.75**	92.5
AM IS+ accuracy	**93.31**	92.94
AM execution time (secs.)	528	643
AM iterations	2.33	2.5
AM classes	18.33	20.11

Table 5 summarizes all of the experimental runs of PIGSCR on both datasets, using the SVD or PCA as a means of feature reduction. The values in the table are computed by averaging the results of 11 runs for the Virginia data (four through fourteen bands used) and 18 runs for the Amazon data (three through twenty). Classification accuracies are reported for both PIGSCR output images that contain only two classes (the DR and IS+ images), corresponding to the validation set containing only two classes. Consider the classification accuracies listed, and notice that using the SVD to reduce the data dimension resulted in higher accuracies than using the PCA for both classification images (DR and IS+) using both datasets.

Considering now just the VA dataset, a comparison between the SVD and PCA using McNemar's test for statistically different classifications as described above showed that the classifications were consistently the same. All of the classification accuracies of the Virginia data are similar, but the accuracies for the SVD are more often higher (seven out of eleven cases). For this particular dataset and classification (forest/non-forest), the subspace spanned by the first few axes of decreasing variance is likely similar to the subspace revealed by the SVD, accounting for the lack of distinction between the two feature reduction methods.

Table 6. Classification Accuracies of Statistically Different Classifications.

bands	class	SVD accuracy	PCA accuracy
6	DR	**95**	91.5
6	IS+	**95.5**	92.5
7	DR	**93.5**	91
7	IS+	**94.5**	92
9	DR	92.5	**95.5**
9	IS+	92.5	**95.5**
10	IS+	**94**	91.5
12	DR	**94**	92
12	IS+	**94.5**	92.5
19	IS+	92	**94**

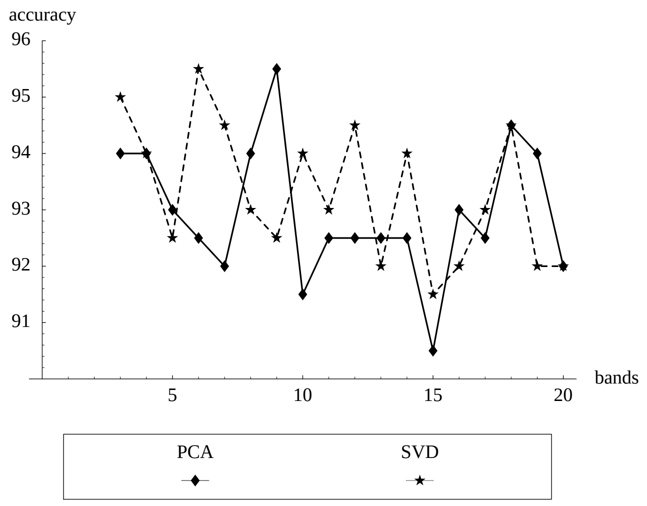


Figure 20. IS+ classification accuracy for AM232.

Applying the same feature reduction methods and classification to the data from the Amazon region, however, demonstrated more separation between the two feature reduction methods. Table 6 lists all classification accuracies for the SVD and PCA AM232 images where the SVD and PCA reduction to the same number of bands resulted in statistically different classifications. In seven out of ten cases, the SVD feature reduction resulted in a higher classification accuracy. Figure 20 contains a graph comparing the classification accuracies of the two feature reduction methods for different numbers of bands kept (singular value or eigenvalue cutoff). Notice that the SVD accuracies are consistently higher than the PCA accuracies, and increasing the number of bands does not usually result in increased accuracy. Of particular interest are the cases where the number of bands is six or nine, where one method drastically outperformed the other regarding accuracy. Figures 21—24 contain subsets of the classification results of AM232 for the SVD and PCA images, using either six or nine bands. Observing Figures 21 and 24, it appears that these images contain more misclassified forested regions due to cloud cover. Also notice that in Figure 23, the most accurate PCA classification, the river running vertically through the image has been incorrectly classified as forest, showing that this classification is not necessarily better than the corresponding SVD classification.

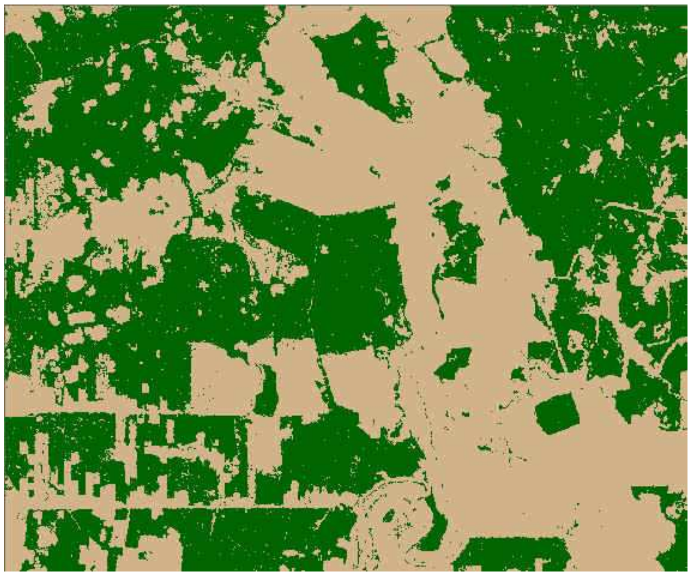


Figure 21. AM232 (Landsat TM path 232/row 67) PCA IS+ classification (6 bands), green = forest, tan = non-forest.

A more subtle advantage of using the SVD over PCA for feature reduction prior to PIGSCR classification is the unexpected execution time savings. Although the time savings in the Virginia classifications was minimal, the execution times for the classification applied to the SVD AM232 image are consistently shorter than execution times for the classification applied to the PCA AM232 image, as shown in Figure 25. An analysis of PIGSCR reveals that execution time can be affected by the size of the image, the number of iterations required, and the number of pure classes. Since the PCA and SVD produce images of the exact same size, the differences in execution time can be attributed to differences in the number of iterations and classes. Notice that the peaks in the execution time graph (Figure

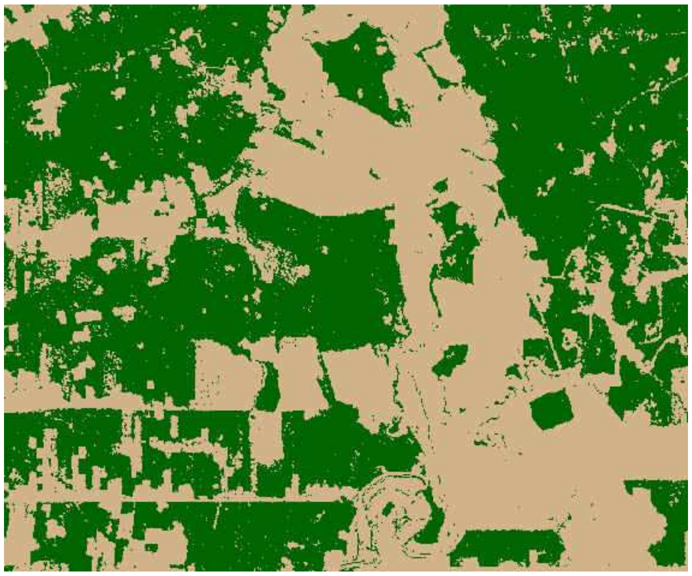


Figure 22. AM232 (Landsat TM path 232/row 67) SVD IS+ classification (6 bands), green = forest, tan = non-forest.

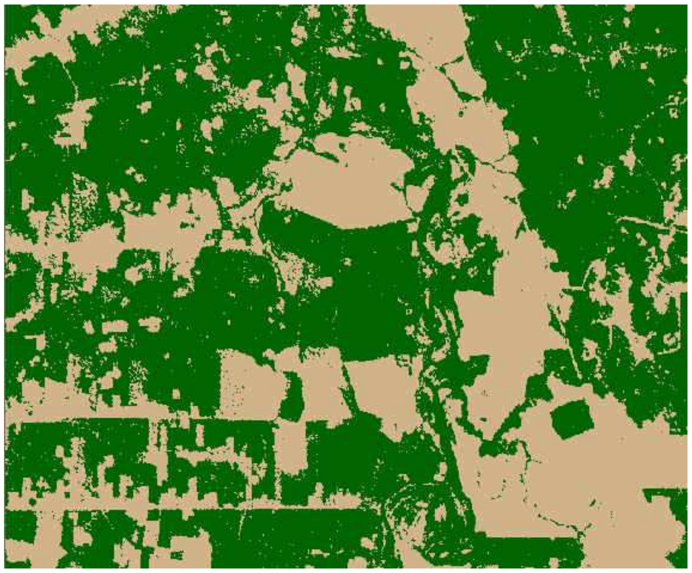


Figure 23. AM232 (Landsat TM path 232/row 67) PCA IS+ classification (9 bands), green = forest, tan = non-forest.

25), correspond to peaks in Figure 26 (number of iterations) and peaks in Figure 27 (number of classes). A small number of iterations and classes does not appear to negatively impact overall classification accuracy. For example, at six bands, classification accuracies were higher using the SVD for feature reduction (and were statistically different classifications than when using PCA), yet the classification on the PCA data required more iterations and resulted in more pure classes. Although this advantage seems tailored to PIGSCR, the implications extend to classification in general, both supervised and unsupervised, as PIGSCR is a hybrid classification algorithm and exhibits characteristics of both.

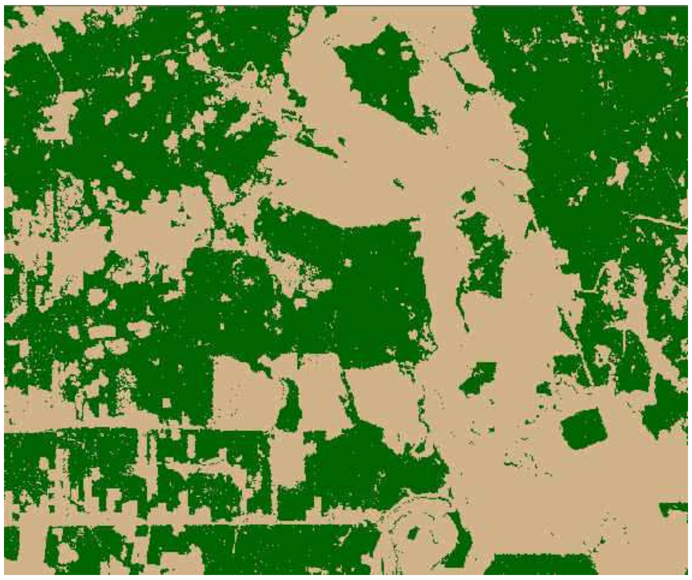


Figure 24. AM232 (Landsat TM path 232/row 67) SVD IS+ classification (9 bands), green = forest, tan = non-forest.

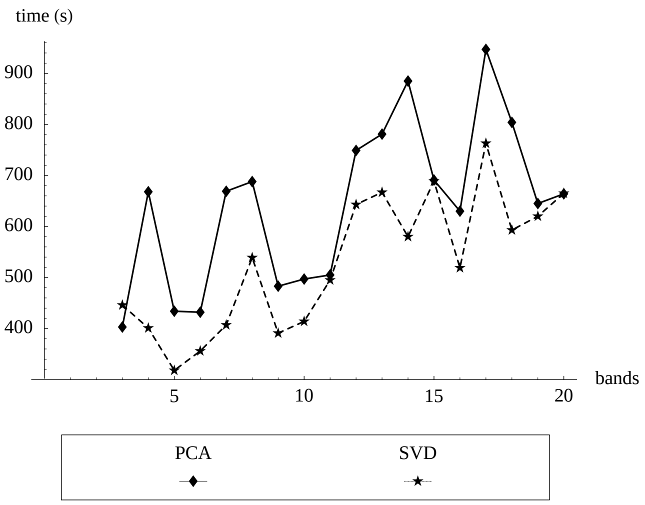


Figure 25. Execution time for AM232.

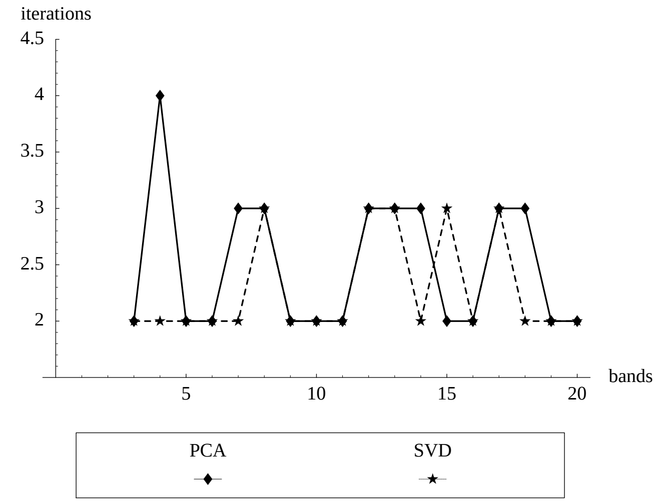


Figure 26. Number of iterations for AM232.

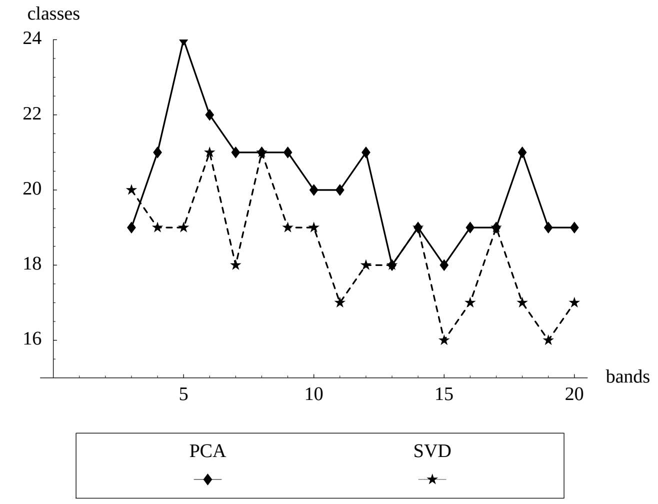


Figure 27. Number of pure classes for AM232.

Chapter 15: SVD CONCLUSIONS

This thesis presents a feature reduction method for remotely-sensed data using the singular value decomposition. This new feature reduction technique was applied to training data from two multitemporal datasets of Landsat TM/ETM+ imagery acquired over a forested area in Virginia, USA and Rondônia, Brazil. PIGSCR forest/non-forest classifications of the Virginia data were five times faster using SVD reduction without affecting the classification accuracy. Feature reduction using the SVD was also compared to feature reduction using principal components analysis (PCA) for both datasets. The highest average accuracies for the Virginia dataset (88.34%) and for the Rondônia dataset (93.31%) were achieved using the SVD. SVD-based feature reduction can yield statistically significantly better classifications than PCA.

This thesis demonstrated the utility of SVD-based feature reduction on images containing fewer than 100 bands, however, SVD-based feature reduction of higher dimensional data (i.e., hyperspectral) should be evaluated in the future. Another area for future exploration with SVD-based feature reduction is classification with increased categorical specificity. With less variability between informational classes, feature reduction based on variance (such as with PCA) is likely to produce a lower quality classification on a reduced dimension image than feature reduction based on SVD. Finally, this thesis demonstrates that applying the SVD to training data in order to reduce the dimension in an entire image produces good classification results. An ideal implementation would apply the SVD to the entire image to reveal the exact basis vectors, not an approximation derived from the training data. A parallel SVD implementation on multiple processors would allow the SVD to be performed on a large image in a reasonable amount of time, likely resulting in greater classification accuracy.

Section III. ADAPTIVE NOISE FILTER.

Chapter 16: BACKGROUND ON NOISE REMOVAL

Hyperspectral images provide a powerful tool as the wave spectrum is finely discretized using hundreds of channels on a scanner. The large dimensionality of a hyperspectral dataset often requires a data transformation such as principal components analysis (PCA) or the singular value decomposition (SVD) to reduce the number of variables, or bands, within an image prior to further processing. Furthermore, these images tend to be noisy as a result of the fine discretization and other factors such as the method of acquisition (using small aircraft at low altitudes). Green et al. first proposed the maximum noise transform (alternately called the minimum noise transform, minimum noise fraction, or MNF) to align a dataset in order of decreasing signal-to-noise ratio (SNR) using an eigenvalue decomposition similar to PCA [38]. Lee et al. equivalently defined the MNF (or noise adjusted PCA) as two PCA transformations, and used the MNF to reduce the noise level in an image [63]. The MNF can be used to reduce noise and the number of dimensions in an image. Reduction in noise in imagery is essential to many remote sensing applications, as Landgrebe has documented the relationship between noise in imagery and classification errors [59]. Furthermore, certain applications require a minimum SNR, for example, estimating foliar biochemical concentrations [86].

Noise can be reduced using a variety of filters defined on the frequency or spatial domains [21]. While certain frequency domain filters (using the Fourier transform) have been shown to be more effective than spatial filters with respect to specific types of noise, a spatial filter such as a median filter can produce similar results and requires significantly less computation [74]. An adaptive filter can alter the size of the filter kernel (spatial domain) or change the frequencies filtered (frequency domain) depending on image characteristics and noise levels. Lennon et al. used an adaptive median filter on data transformed to MNF coordinates [64], and Pok et al. vary the kernel size between three and five depending on the detected noise in a particular window in a three-band image [72]. King et al. present an adaptive frequency domain filter used on medical imagery [56].

The properties of the MNF are well suited to an adaptive filter, yet adaptive spatial filtering is not commonly used on MNF transformed data, although the idea was proposed by [64]. Typical data processing using the MNF truncates the data, resulting in loss of signal, uses all bands in the MNF coordinate system without noise removal, or applies a spatial convolution with a uniform kernel size across all bands despite all bands having drastically different SNRs. This thesis introduces an algorithm for an adaptive median filter applied to data transformed using the MNF in which the filter support size varies with noise. To demonstrate the effectiveness of this technique, a real dataset is filtered using the algorithm presented in this thesis.

Chapter 17: MAXIMUM NOISE FRACTION

Data transformations such as PCA and SVD transform an image to a new coordinate system without taking factors such as noise into consideration [38]. PCA uses the eigenvectors (V) resulting from an eigen decomposition:

$$\Sigma = V\Lambda V^{-1},$$

where Σ is the covariance matrix of the image and Λ is a diagonal matrix containing the eigenvalues corresponding to V, as a new coordinate basis for the image. This transformed image has the property that each successive band is aligned along an axis of decreasing overall variance in the original image; that is, as the component number increases, the variance within the component decreases. When PCA is used for data reduction, ideally these higher order bands with decreasing variances are not necessary to represent the majority of the original image, and these components can be removed, resulting in a data reduction. Unfortunately with datasets that are particularly noisy (the case with hyperspectral data), the first few components are not sufficient to represent the image as they capture much of the noise as well as the signal. The MNF is similar in spirit to PCA with the additional quality that it considers image noise when selecting a new coordinate system. While the PCA aligns the axes along directions of the maximum variance in the original image, the MNF aligns the axes along directions of the maximum SNR.

Theoretically, the MNF orders the data along the axes of maximum SNR using the eigen decomposition

$$\Sigma_S \Sigma_N^{-1} = V\Lambda V^{-1},$$

where Σ_S is the covariance matrix of the signal, Σ_N is the covariance matrix of the noise, V is an (orthogonal) matrix containing the eigenvectors of $\Sigma_S \Sigma_N^{-1}$ (assumed to be a normal matrix), and Λ is a diagonal matrix containing eigenvalues that correspond to V. V provides the basis for the transformed dataset. In practice, Σ_S and Σ_N are unknown and must be estimated from the data [38]. Σ_S is generally taken to be the covariance matrix of the image, and Σ_N can be estimated using various procedures [38][63]. The eigenvalues contained in Λ are the estimated variance of the signal (σ_S) divided by the estimated variance of the noise (σ_N), and therefore the diagonal element λ_b in Λ is an approximation for the SNR of band b in the transformed image.

Chapter 18: ADAPTIVE FILTER

The MNF is commonly used in remote sensing for data reduction and noise removal. The MNF of an image can be truncated while still preserving most of the information within the image, which is especially useful in the hyperspectral image processing domain as images contain hundreds of highly correlated bands and noise. The higher order bands that are truncated commonly contain very low SNRs, and truncating the MNF can have the added effect of eliminating much of the noise without losing much signal. Determining the precise location to truncate the MNF is problematic, and a judgement call is often made by looking at a plot of the eigenvalues relative to the band number and determining where this eigenvalue curve begins to approach an asymptote ($\lambda = 1$). The SNRs estimated by the MNF described by [38] approach this asymptote because the signal is estimated by the variance of the image, which includes noise. In practice, this truncation is performed as a means of reducing the overall noise within the image, but this method does not fully take advantage of the properties of the MNF. If the truncation includes too many bands, too much noise is left in the image, and if the truncation includes too few bands, useful signal may be excluded from the resulting image. A likely scenario would be that truncation includes noise in the bands that are kept while discarding good signal with the higher order bands that are discarded.

Green et al. suggest that with low SNR bands, all values can be replaced with the mean of the band and the MNF image can be retransformed to the original subspace, resulting in a less noisy image [38]. This is an example of a rather extreme mean filter. Another approach to reducing the noise in an image is to apply a small (typically a three by three window) spatial filter such as a mean or median filter. However, applying a filter uniformly to all bands within the MNF will not take advantage of the specific ordering of the bands. Bands with lower SNRs might benefit from a filter with a larger window, while bands with high SNRs require little or no filtration. Bands with low SNRs have comparatively low signal relative to noise, yet may have enough signal to warrant smoothing of the noise. A large filter will degrade that signal, but will likely affect the noise more because of implicit spatial correlations present in the signal, resulting in a greater signal relative to noise [80].

Spatial median filters work by decreasing the variance within a small window (kernel) by assigning a pixel the median value of the surrounding pixels. For example, using a 3×3 window, a median filter would assign a pixel the median value of itself and its eight immediate neighbors (top, left, right, bottom, and four diagonal locations). As geographic data are highly correlated, the variance of the signal within such a window should be small and noise should be random and not correlated within a neighborhood, making a large variance probable. With the assumption that the variance of the noise is larger than that of the signal in these small windows, a spatial filter such as a median filter will preserve most of the signal while eliminating much of the noise. A filter with a larger window has a more dramatic smoothing effect over a filter with a small window, resulting in a larger SNR at the expense of the signal. A median filter has the property of preserving original values unlike a mean filter.

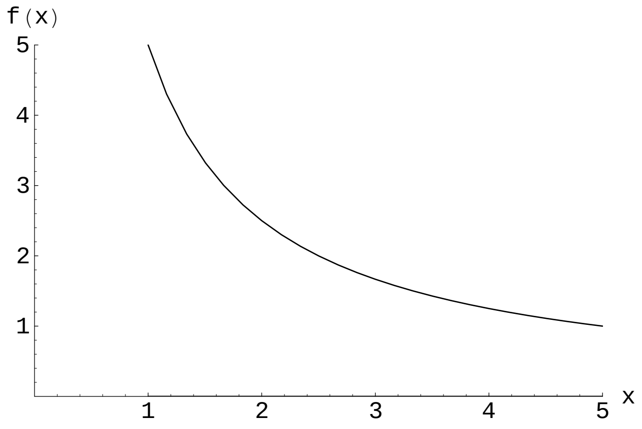


Figure 28. Typical MNF eigenvalue curve shape.

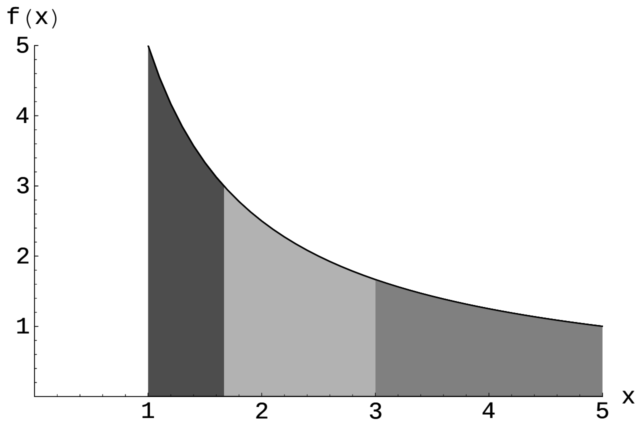


Figure 29. Dividing area under curve into bins.

18.1 Determining Filter Kernel Size

The properties of the MNF and the relationship between large filter kernels and increased noise reduction can be used to construct an adaptive filter with increased kernel sizes for lower SNRs. The MNF is ordered such that for any two bands numbered m and n (assume $n > m$), $\mathrm{SNR}_m \geq \mathrm{SNR}_n$. Recall that the eigenvalues associated with the MNF are estimates of SNR, meaning $\lambda_m \geq \lambda_n$. Therefore, the size K of the filter kernel for band m should not be greater than that for band n, $K_m \leq K_n$. Similarly, the same size filter should be applied to bands with the most similar eigenvalues. Consider the shape of the typical MNF eigenvalue curve, shown in Figure 28. The first few bands with the largest decrease in slope should be grouped together in smaller groups than the last bands with very similar small negative slopes. In order to divide the bands into bins in this manner, the area underneath the eigenvalue curve can be divided evenly into a number of bins corresponding to the number of different sized filters to be applied, as shown in Figure 29. The three colors represent three different bins and three different kernel sizes. Since the exact function $f(x)$ is unknown, $f(x)$ will be approximated by $C(x)$, a function that interpolates the points $(x_i, f(x_i))$, $i = 1, \ldots, B$, where B is the number of points. A formal algorithm for the adaptive filter (AF) is given below.

```
Algorithm AF($M, \Lambda, nb$)
input/output:
$M$ (image transformed to MNF coordinates,
filtered upon exit)
input:
$\Lambda$ ($B$ eigenvalues where $B$ is number of bands in $M$)
$nb$ (number of bins to use)
1  begin
2    approximate $f(b) = \lambda$ with $C(b) = \lambda$
3    $area := \int_1^B C(b)db$
4    $area\_per\_bin := \frac{area}{nb}$
5    $sarea := 0$
6    for $i := 1$ step 1 until $B - 1$ do
7      begin
8        $sarea := sarea + \int_i^{i+1} C(b)db$
9        $bin :=$ ceiling $\left(\frac{sarea}{area\_per\_bin}\right)$
10       $kernel_i := 2 \cdot (bin - 1) + 1$
11     end
12   $kernel_B := kernel_{B-1}$
13   for $i := 1$ step 1 until $B$ do
14     apply $kernel_i \times kernel_i$ median filter
       to $band_i$ in image $M$
15 end
```

Calculating the area under the curve will require a function to approximate the eigenvalues as a function of band number, as indicated in line 2 of the above algorithm. A description of Hermite splines, which are recommended given their suitability to this particular application, is included in Chapter 18.2. The assignment of bins occurs in line 9, and warrants further explanation. Starting with the first band, the area under the curve is calculated as

$$\int_1^2 C(b)db.$$

The total area under the curve through band i is therefore

$$\int_1^{i+1} C(b)db.$$

The results of previously calculated integrals are stored in *sarea* to prevent redundant calculations. Taking the ceiling of the cumulative area under the curve divided by the area per bin results in bands one through $B - 1$ being placed in bins one through nb, and band B is placed in the same bin as $B - 1$. Line 10 continues with the conversion of a bin number to the size of a spatial filter kernel that corresponds to bin number. The bands in bin one should have no filter (equivalent to a kernel of size one) applied, and the bands in bin two should have a 3×3 filter applied.

This approach is valid for convex eigenvalue curves that are similarly shaped to Figure 28. In the same way that PCA aligns each band along the direction of maximum remaining variance, the MNF aligns each band along the direction of maximum remaining SNR, making convexity of the eigenvalue curve a reasonable assumption. The properties of the MNF dictate that the eigenvalue function is strictly decreasing, but in the event that the eigenvalue function is not convex, dividing up the area under the curve of the derivative of the function will group the most similar eigenvalues and their corresponding bands together. Consider finding the area under the curve of the derivative:

$$\int_a^b f'(x)dx = f(b) - f(a)$$

according to the fundamental theorem of calculus. This calculation requires no approximation of the function as the actual functions' values can be used. Because the eigenvalue function is monotonically decreasing, the area underneath the curve will be negative, and therefore the area will be negated to produce a positive result necessary for bin determination. The above algorithm is modified to produce the following adaptive filter with derivative (AFD) algorithm using the area underneath the curve of the derivative to determine the location of bins.

```
Algorithm AFD($M, \Lambda, nb$)
input/output:
$M$ (image transformed to MNF coordinates,
filtered upon exit)
input:
$\Lambda$ ($B$ eigenvalues where $B$ is number of bands in $M$)
$nb$ (number of bins to use)
1   begin
2     $area := \Lambda(1) - \Lambda(B)$
3     $area\_per\_bin := \frac{area}{nb}$
4     $sarea := 0$
5     for $i := 1$ step $1$ until $B - 1$ do
6       begin
7         $sarea := sarea + \Lambda(i) - \Lambda(i+1)$
8         $bin :=$ ceiling $\left(\frac{sarea}{area\_per\_bin}\right)$
9         $kernel_i := 2 \cdot (bin - 1) + 1$
10      end
11    $kernel_B := kernel_{B-1}$
12    for $i := 1$ step $1$ until $B$ do
13      apply $kernel_i \times kernel_i$ median filter
        to $band_i$ in image $M$
14  end
```

Finally, either of the above variations on the adaptive filtering algorithm may be used on a particular range of bands. For example, if prior knowledge or analysis of the MNF transformed dataset indicates that there is no usable signal beyond a specific band, the MNF image can still be truncated and filtered adaptively. The value of B would be changed from the total number of bands in the image to the number of bands desired after truncation. This is different from simply truncating the MNF because the bands that are kept would be filtered to decrease the noise, and the number of bands kept could be larger to ensure that very little signal is lost in the truncation.

18.2 Hermite Splines

The filtering algorithm requires a function that approximates the eigenvalue curve generated by the MNF. Cubic splines are piecewise cubic polynomials that produce a visually appealing curve and interpolate a given set of points. In particular, Hermite cubic splines have only one continuous derivative (standard cubic splines have two) and produce a monotone cubic spline curve interpolating a monotonic function, rendering this type of spline ideal for interpolating the monotonic SNR curve. The Hermite cubic spline $C(x)$ is composed of $2n$ basis functions, $c_i(x)$, $\hat{c}_i(x)$, $i = 1, \ldots, n$, where n is the number of interpolation points. The function

$$C(x) = \sum_{i=1}^{n} y_i c_i(x) + d_i \hat{c}_i(x)$$

interpolates the points (x_i, y_i), $i = 1, \ldots, n$ if

$$\begin{aligned} c_i(x_i) = 1, c_i(x_j) = 0, &\quad j \neq i, \\ \hat{c}_i(x_j) = 0, &\quad \text{for all } i, j. \end{aligned}$$

Furthermore,

$$\begin{aligned} c_j'(x_i) = 0, &\quad \text{for all } i, j, \\ \hat{c}_i'(x_i) = 1, \hat{c}_i'(x_j) = 0, &\quad j \neq i, \end{aligned}$$

making

$$C'(x_i) = d_i.$$

Only x_i, y_i, and d_i, $i = 1, \ldots, n$ are required to define a Hermite cubic spline, and the d_i are chosen to make $C(x)$ monotone (theoretically always possible for monotone data y_i). Refer to [52] for a more detailed description of Hermite cubic splines including definitions of the basis functions $c_i(x_i)$, $\hat{c}_i(x_i)$.

The derivative and the definite integral of Hermite cubic splines can be easily obtained as the cubic polynomials (and basis functions) are easily differentiated or integrated analytically. Included in [52] is a set of subroutines designed to define, evaluate, and integrate Hermite cubic splines, `PCHEZ`, `PCHEV`, and `PCHQA`, respectively. `PCHEZ` defines continuous derivatives, d_i, that result in a visually appealing function, `PCHEV` evaluates the function and the derivative at a set of points, and `PCHQA` returns the definite integral of the function between two points, a and b.

Chapter 19: AF STUDY AREA

The study area (as described by [91]) is located in the Appomattox Buckingham State Forest in Virginia, USA. Three Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) 224-band, low-altitude flight lines were acquired in the winter of 1999 and ranged from approximately 400-2500nm (10nm spectral resolution) with 3.4m spatial resolution (van Aardt and Wynne 2007). The AVIRIS data were geometrically and radiometrically corrected (to level 1B at-sensor radiance, units of microwatts per square centimeter per nanometer per steradian) by the Jet Propulsion Laboratory (JPL; Pasadena, California, USA). The three flight lines used for this study were registered (8-12 control points per flight line) to an existing 0.5m orthophoto of the area. Resampling resulted in root mean square errors (RMSE) ranging between 0.23 and 0.24 pixels. Training data consisted of 142 field collected locations surrounded by homogeneous areas of single pine species with differentially corrected global positioning system (GPS) coordinates. Three pine species, loblolly (*Pinus taeda*), shortleaf (*Pinus echinata*), and Virginia pine (*Pinus virginiana*), with 64, 30, and 48 locations, respectively, were collected in August of 1999. The image (shown in Figures 40–43 and hereafter referred to as ABSF) contains various tree stands that include the three species of pines listed above, hardwoods, and mixed (evergreens and hardwoods).

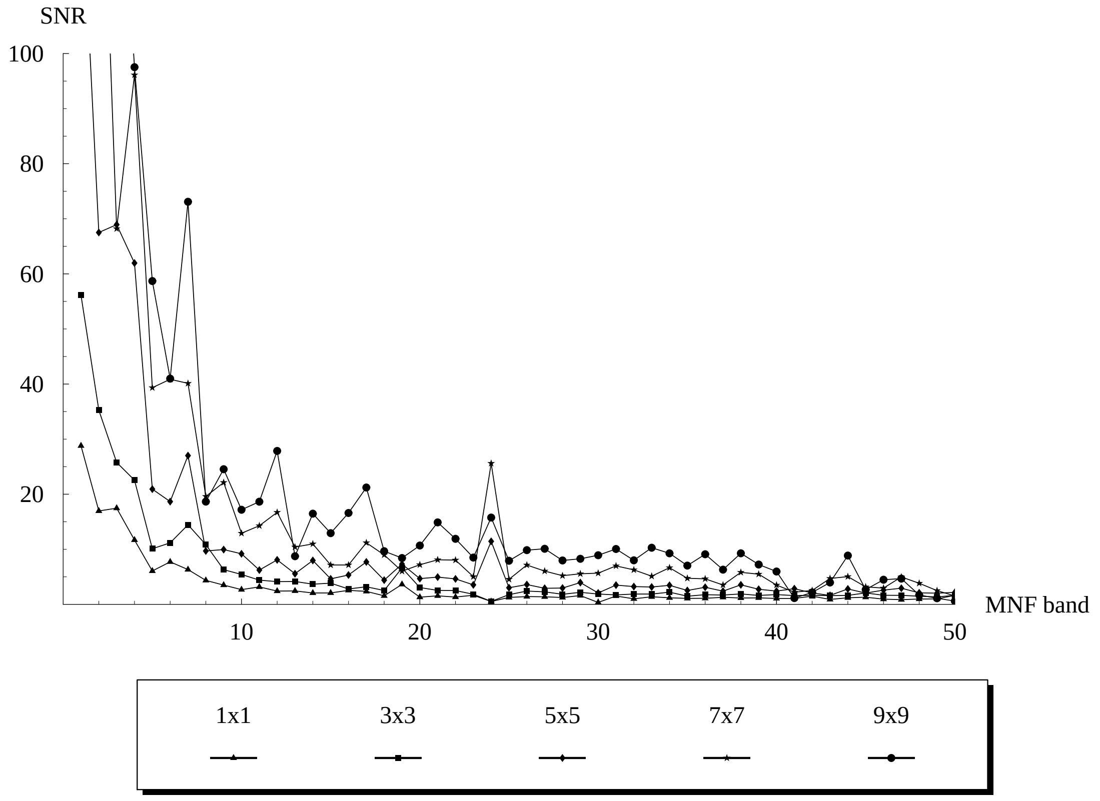


Figure 30. SNR estimates of MNF bands 1–50 unfiltered (1 × 1) and after 3 × 3 through 9 × 9 median filters applied.

Chapter 20: SNR ESTIMATION

To support the conclusion that the adaptive filter is improving the image, evidence is provided to show that the filter is reducing the noise without drastically reducing the signal, effectively improving the SNR. Estimating the SNR in an image is nontrivial, as most noise estimates are variance-based and the naturally occurring variance within an image will lead to overestimated noise. In order to minimize the impact of signal variance on noise estimates, homogeneous portions within an image are identified for noise estimation. Curran and Dungan proposed SNR estimation of AVIRIS images using a homogeneous line within an image [27]. Gao proposed an alternative method that does not require identifying homogeneous regions within an image, but instead divides the entire image into small blocks and uses local standard deviations to estimate the standard deviation of the noise for the entire image [35]. Smith and Curran found both methods effectively estimated SNRs in AVIRIS images, although both methods overestimated SNR [86]. As the study area used for this thesis lacks clearly defined homogeneous regions, Gao's whole image method of SNR estimation will be used, as it does not require identification of homogeneous regions and it can be easily automated [35].

Gao's method assumes that within a small block (4 × 4 or 8 × 8) the local standard deviation is either low because of noise and a small amount of natural variance or high because the block contains edges, etc. An image band is broken into small blocks of equal

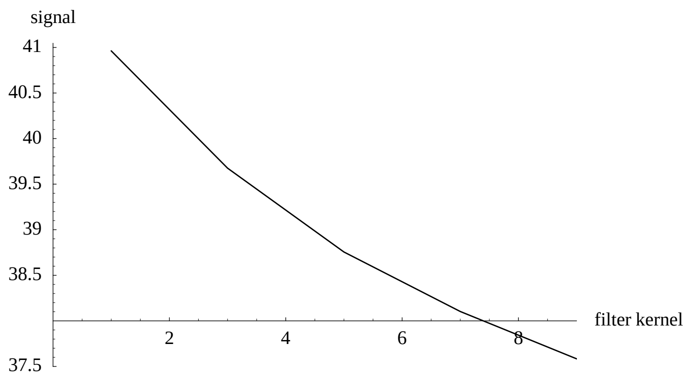


Figure 31. Signal estimate of MNF band 1 with various sized median filters applied.

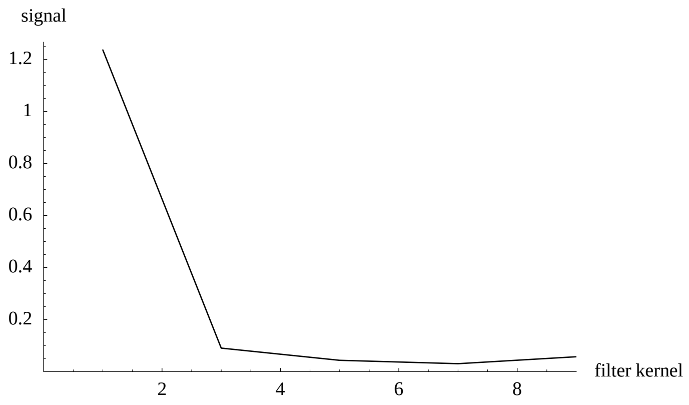


Figure 32. Signal estimate of MNF band 50 with various sized median filters applied.

size, and the local standard deviation is calculated for each block. A histogram is created from the local standard deviations, and the most frequently occurring standard deviation provides a reasonable estimate of the standard deviation of the noise for the entire band. Gao generated data with a known amount of noise and showed that this method accurately estimated the noise [35].

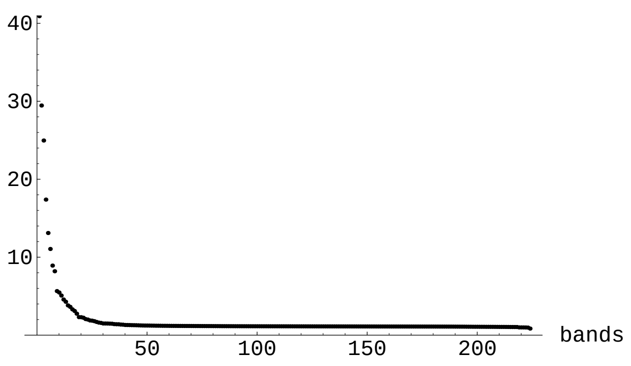


Figure 33. Study area MNF SNRs (eigenvalues).

In this SNR estimation method, the signal in AVIRIS images is generally estimated using the mean values of each band, however the mean values of MNF transformed bands do not have the same magnitude as the mean values in the original coordinate system. The original digital numbers within the image are all positive, and MNF digital numbers can be positive or negative, resulting in mean values that are much lower in the resulting MNF coordinate system. In order to study the effects of filtration on SNRs in MNF coordinates, an estimate suggested by Schowengerdt is used:

$$SNR = \frac{\sigma_S^2}{\sigma_N^2},$$

where σ_S^2 is the variance of the signal and σ_N^2 is the variance of the noise [80]. The variance of the signal is estimated by the variance of the entire image, and the variance of the noise is determined using the method described previously. Note that this method is not being suggested to estimate SNRs in order to evaluate whether an image can be used for a certain application, or to compare an image to other images whose SNRs are estimated using other procedures. This estimation of the signal, noise, and SNR is needed only to compare unfiltered and filtered data to provide evidence that the SNRs are improving as a result of filtration.

The behavior of the SNR was evaluated using the methods described above, and the test image was transformed to MNF coordinate space and filtered using a 3×3, 5×5, 7×7, and 9×9 median filter. Notice in Figure 30 that as the filter size increases, the SNR also increases compared to the unfiltered image (1×1). Also, notice in Figures 31 and 32 that since the magnitudes of the signal are very different in bands one and 50, degrading the signal in the first band has more impact on the total image signal degradation than removing some signal in the 50^{th} band.

Chapter 21: AF RESULTS

The AF, AFD, truncated AF, truncated AFD, and uniform 9×9 median filters were applied to the MNF of ABSF. Results for the 9×9 filter are reported as the 9×9 filter produced the best classification accuracies for all uniform spatial filters. Further evidence of improved image quality as a result of adaptive filtering is provided in a classification of ABSF to identify loblolly, shortleaf, and Virginia pines. An advantage of the technique used (discriminant analysis) is that individual bands are selected by the method, showing that high order noisy MNF bands contain signal that impacts applications such as classification.

21.1 Bin Creation

The adaptive filtering algorithms (using the approximation of the curve and using the derivative of the curve to establish bin locations) described above were applied to ABSF. Figure 33 shows the eigenvalues (SNRs) generated by the MNF. Figures 34 and 35 show the resulting eigenvalue function and derivative curves (for the entire image and for the image truncated to 80 MNF bands) approximated by Hermite splines, with the area underneath the respective curves divided into five equal partitions, representing median filter kernels of size one through nine.

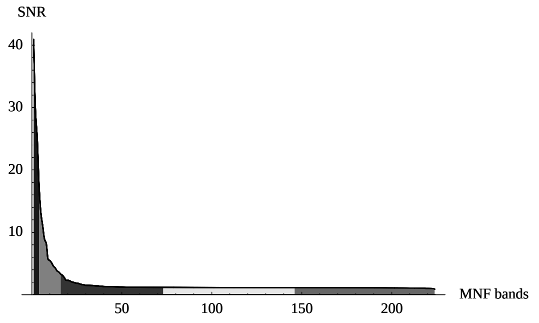


Figure 34a. Bins generated by AF on study area.

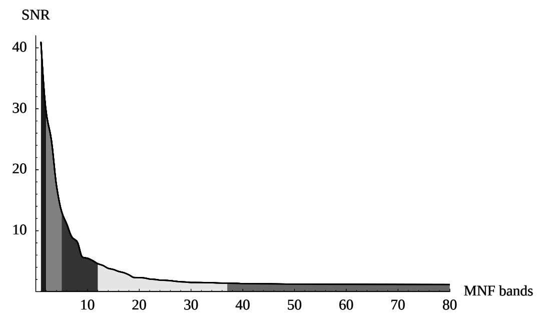


Figure 34b. Bins generated by AF (truncated to 80 MNF bands) on study area.

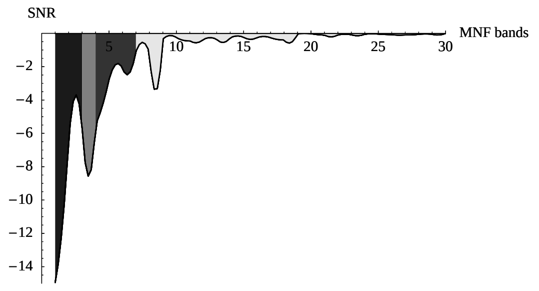


Figure 35a. Bins generated by AFD on study area, bands 1–30 shown.

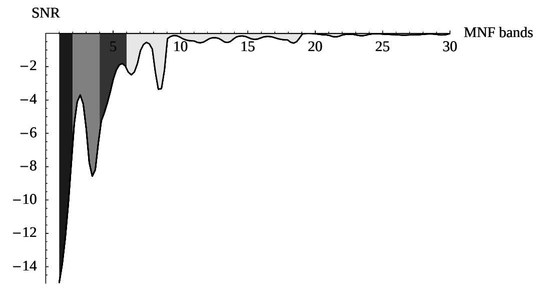


Figure 35b. Bins generated by AFD (truncated to 80 MNF bands) on study area, bands 1–30 shown.

21.2 Classification

A discriminant analysis was used to generate classification functions for each filtered and unfiltered dataset using SAS$^{(R)}$ 9.1.3 [79]. First the `STEPDISC` procedure was used to identify bands that contribute to the discrimination between the three pine species. The `STEPDISC` procedure (using stepwise selection) starts with no variables in the model. The variable that contributes most to species discrimination enters the model if the significance level of an F-test is above the specified threshold. In each subsequent step, the variable within the model that contributes least to discrimination is first considered for removal (if the significance level of an F- test is beyond the specified threshold to leave the model). If no variable is removed, of the remaining variables not included in the model, the variable that best contributes to the discrimination of the model is added, if it meets the criterion to enter. At each step, either one variable leaves the model or one variable enters the model until the procedure terminates. The procedure terminates when no more variables are eligible to enter or leave the model, or a maximum number of steps is reached.

`DISCRIM` generates a discrimination function (classification) when given a set of quantitative variables and corresponding classifications for each observation. Classification accuracies are determined by using the generated discriminant function to classify each observation, which is a biased test. A better measure of accuracy is the cross validation accuracy, where each observation is classified using the classification model derived from the other observations (not including itself).

Table 7. Results of stepwise discriminant analysis and discriminant analysis on study area using various filtering schemes in both original and MNF coordinates (Legend: CV Acc. = Cross Validation Accuracy (%), Cls. Acc. = Classification Accuracy (%)).

Filter	CV Acc.	Cls. Acc.	α-level	# Bands	Bands
MNF coordinates					
none	85.92	88.03	.01	9	1,2,3,5,7,13,16,18,173
AF	91.55	93.66	.001	13	1,2,6,7,9,10,14,16,19,36,44,52,148
AFD	98.59	100.00	. 00001	11	5,10,12,16,18,19,34,36,45,48,53
Trun. AF	98.59	98.59	.0001	10	2,7,14,16,19,29,36,44,45,53
Trun. AFD	98.59	100.00	. 00001	11	5,10,12,16,18,19,34,36,45,48,53
9x9	97.89	100.00	. 00001	11	5,10,12,16,18,19,34,36,45,48,53
original coordinates					
none	76.76	79.58	.0001	4	19,24,116,213
9x9	91.55	92.25	.00001	4	3,19,27,164
9x9	90.85	92.96	.0001	5	3,19,26,27,164
AF	83.10	83.80	.001	5	19,26,27,126,158
AFD	91.55	92.25	.00001	4	3,19,27,164
AFD	87.32	92.25	.0001	6	4,9,19,23,27,44
9x9	95.07	97.89	.001	11	3,19,23,37,105,120,140, 144,147,183,200
AF	92.25	97.89	.01	14	11,16,19,26,27,30,43,120,158, 161,183,194,197,222
AFD	92.96	96.48	.01	10	4,9,19,23,27,40,41,42,164,218
AFD	93.66	95.07	.001	8	4,9,19,23,27,41,44,164

Table 8. Results of stepwise discriminant analysis and discriminant analysis on study area using various truncated filtering schemes in original coordinates (Legend: CV Acc. = Cross Validation Accuracy, Class Acc. = Classification Accuracy).

Filter	# MNF Bands Kept	CV Acc. (%)	Class Acc. (%)	α-level	# Bands
none	40	88.03	92.25	.1	12
none	60	84.51	88.03	.1	14
AF	80	95.78	97.89	.01	12
AFD	80	93.66	94.37	.01	10
none	60	73.24	77.47	.001	4
AF	80	86.62	89.45	.0001	4
none	40	83.10	85.92	.01	8
AFD	80	94.37	95.07	.001	7

The classification results in this thesis were generated using `STEPDISC` with stepwise selection, and the variables (bands) identified were used to generate the discriminant function using `DISCRIM`. The default value of .15 was used as the threshold for entering and leaving the model in `STEPDISC`, and the default assumption of normality was used to generate the discriminant function in `DISCRIM`. In order to select different numbers of hyperspectral bands, the α-level for the F-test was varied between .1 and .00001 for different classifications reported in Tables 7 and 8. The data and discriminant analysis procedure are identical to those used to obtain maximum cross validation classification accuracies of roughly 85% using a maximum of 10 bands in [91].

The first set of entries in Table 7 correspond to classifications applied to transformed data in the MNF coordinate system. Accuracies for MNF data with no filter applied are recorded for comparison purposes. The second set of Table 7 entries are classification accuracies and parameters using data that were filtered in the MNF coordinate system and then inverse transformed to the original coordinate system. The classifications in Table 8 are applied to data that are transformed to MNF space, truncated or truncated and filtered to remove noise, and inverse transformed. The unfiltered MNF data are truncated at 40 or 60 MNF bands because these thresholds appear to be located near where the eigenvalue curve approaches one (see Figure 33). As more noisy bands are included, the accuracies of the classification of truncated MNF data decrease, and therefore results are not reported for the inverse of MNF data truncated at 80 bands, corresponding to the adaptive filters applied to MNF truncated 80 band images.

Chapter 22: AF DISCUSSION

Classification results in Table 7 indicate that using any of the mentioned variations of adaptive filtering ultimately improves the classification accuracy over not using any filtering method, indirectly indicating that the adaptive filter has improved the image quality, enabling better classification results. Furthermore, considering the first six table entries corresponding to classification in the MNF axes, the more accurate classifications make use of high order MNF bands. The least accurate classification (applied to unfiltered data) primarily included bands from the first 20 MNF bands, whereas the more accurate classifications consistently used higher order bands in the 30–55 range. Many of the bands identified to be important for pine species discrimination are consistent across filtering schemes, but are not present in the classification of the unfiltered dataset. These results support the hypothesis that there is signal in higher order MNF bands that can be important for applications such as classification, and that applying a filter to reduce the levels of noise can make the weak signal in these bands usable.

Recall that the MNF maximizes SNR in each band where the signal is estimated by the variance of the image, making the transformation similar to PCA, which aligns the data along axes of maximum variance. In applications such as this where the goal is discrimination between pine species, higher order MNF bands (and PCA bands) might be expected to play a vital role as the variance between pine species is likely small. Unfortunately these high order bands in the MNF transformation that have little signal (variance) have relatively large amounts of noise. Simply truncating the MNF to remove this noise will result in also removing signal.

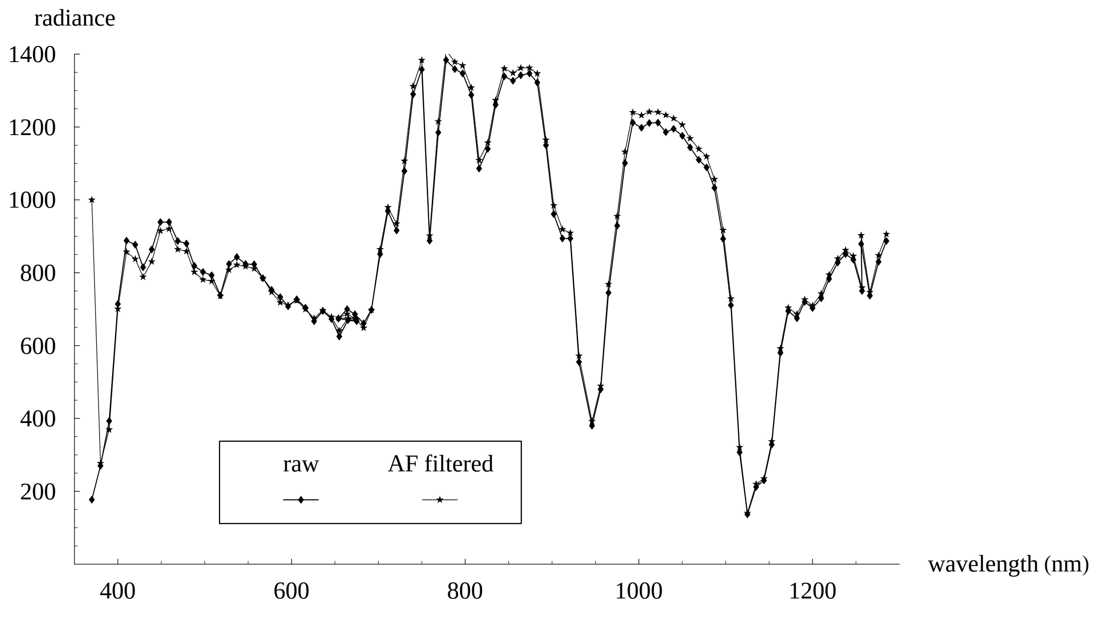


Figure 36. Spectra of loblolly pine training pixel before and after filtering.

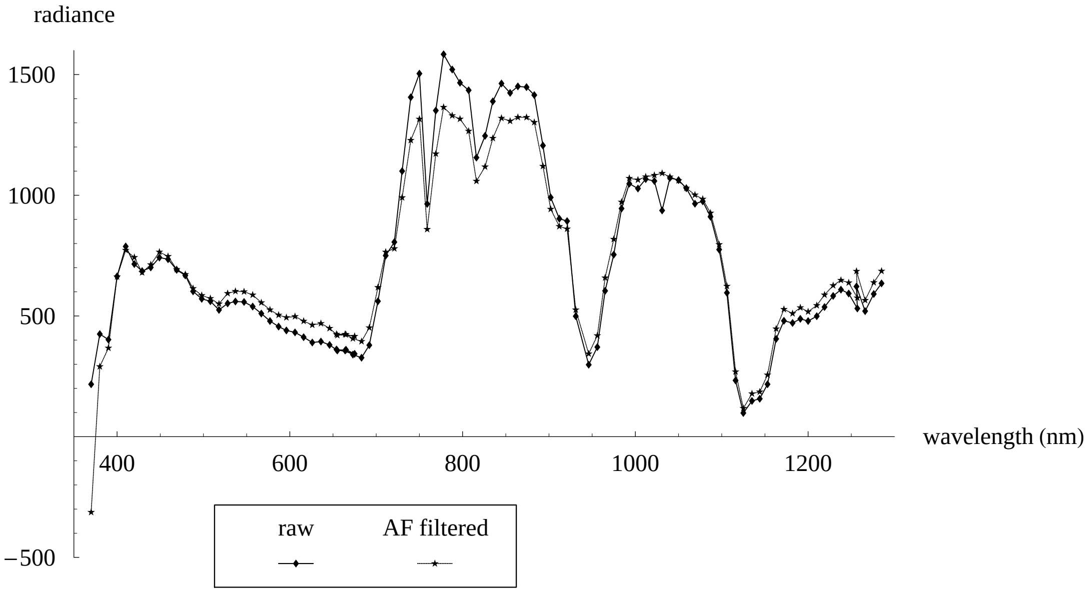


Figure 37. Spectra of short leaf pine training pixel before and after filtering.

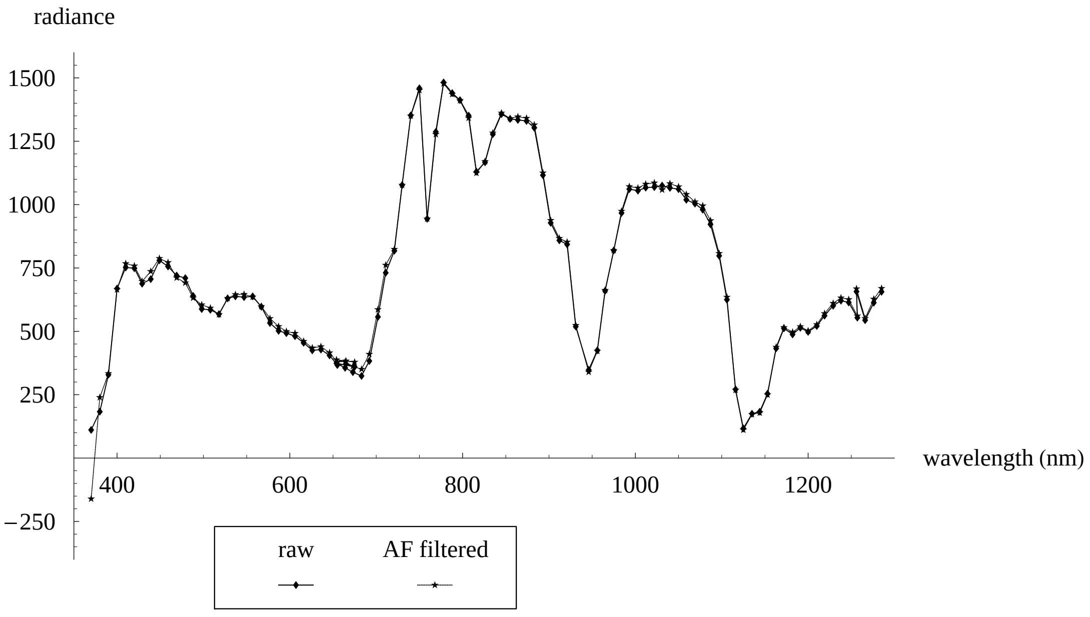


Figure 38. Spectra of Virginia pine training pixel before and after filtering.

The second set of entries in Table 2 corresponding to classification results in the

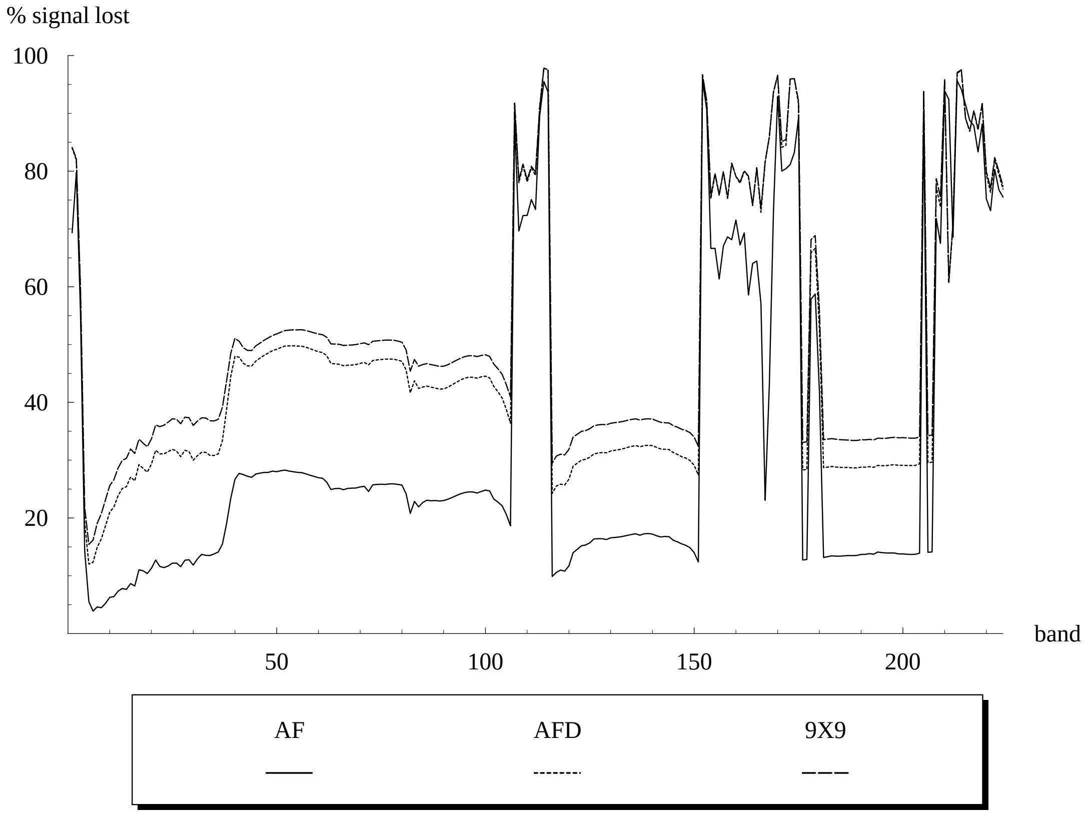


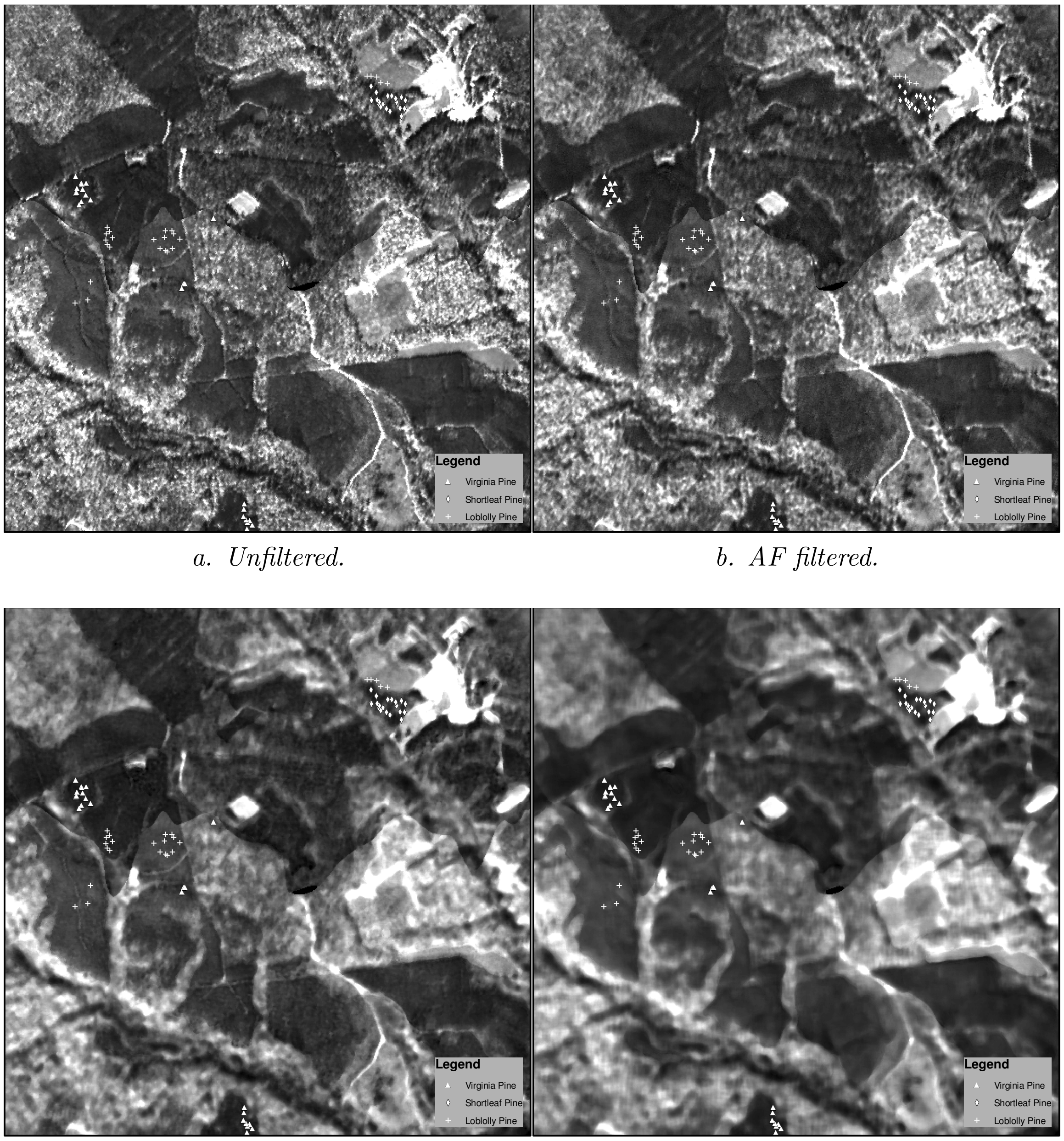
Figure 39. Percentage of signal lost by AF, AFD, and 9 × 9 median filters.

original coordinate system show that the improved MNF image quality is consistent when the image is inverse transformed. This set of classification accuracies are not as high as those corresponding to the MNF coordinate system, even when the same number of bands is used to build each of these models. Although the numbers of bands used in these classifications are similar to those used on the MNF coordinate data, the MNF coordinate data still results in better classifications. However, some classifications require that data not be transformed (e.g., spectral angle mapper using spectral libraries [14]), and the results in Table 1 demonstrate that the MNF-based adaptive filtering methods are relevant to images that are inversely transformed from MNF coordinate space. Note that in order to use a spectral library for classification, the spectra should be preserved by the transformations and filtering. Figures 36, 37, and 38 contain spectra for a loblolly pine training point, a short leaf pine training point, and a Virginia pine training point, respectively. The shapes of the spectra do not appear to be significantly altered in Figures 36 and 38, but there is some alteration of the spectrum in Figure 37. These MNF-based adaptive filtering methods may potentially be used prior to classification and species identification using spectral libraries if spectra are not significantly altered. In cases where spectra are significantly altered, certain applications such as vegetation indices would be hindered. Similar spectra altering effects occur when using comparable filters in highly heterogeneous areas. Note in Figures 37 and 38 that negative radiance values can results from the combination of data transformations and filtering. Negative radiance values were uncommon overall in the experimental data, but were common in noisy bands.

For this particular dataset, the AFD usually produced better classifications (indicated by classification accuracy and cross validation classification accuracy) than the AF, however, for other applications, the AF may be the better choice. The following discussion shows that preserving image quality (signal) and achieving high classification accuracy are *not* equivalent — AF has better image quality (more preserved signal), but worse classification/cross validation accuracy, than AFD. Referring back to Figures 34 and 35, recall that using the derivative curve to determine bin placement resulted in larger spatial filters being applied to more bands within the image. In fact, the classification results for the 9×9 filter are consistently similar to results obtained using the AFD filter because in this case, the filters are very similar. As stated previously, the low order bands containing most of the signal (variance) of the entire image likely do not greatly aid discrimination between spectrally similar pine species. These filters are substantially degrading the signal in the image, as shown in Figure 39. Because the first MNF bands contain the majority of the signal, applying a filter with such a large window drastically degrades the signal of the overall image. Figure 39 shows that while the AF filter decreases the variance of the original image by around 20% for most bands, the AFD and the 9×9 filters decrease the variance by around 40% for many bands. There is a noticeable difference in the signal (variance) degradation between the AF, AFD, and 9×9 filters, indicating that using the AFD and, especially, a large uniform filter substantially reduces the image signal.

Further evidence of image quality can be observed qualitatively by viewing and comparing the filtered images to the original image and the unfiltered MNF image. Figure 40 contains band 27 of the image in the original coordinate system, a band that was used for the pine species discrimination in many of the images. While Figure 40b (the inverse of the MNF ASF image) does not appear as crisp as the original image, the texture is much greater than that of either Figure 40c (ASFD) or 40d (9×9). Figure 41 compares the four filtering schemes applied to band 5 of the MNF image, providing insight into the obvious signal degradation in band 27. While the AF image (Figure 41b) is relatively crisp (3×3 filter used), the AFD image (7×7) and the 9×9 image have substantially degraded signal, resulting in substantially degraded signal for the overall image (band 5 contains more of the image's signal than each subsequent band). A much higher order band, band 36, is shown in Figure 42. This band was selected for each MNF filtered classification, but was not part of the less accurate classification of the unfiltered image. Although it is possible to distinguish a small signal in Figure 42, the noise clearly dominates this band. The spatial filters are reducing the signal of the image, but importantly reduce the noise to a level that reveals variance between individual portions of the image. The areas marked by ground truth can be spectrally distinguished in this band (which is important to the classification) even if the signal does not appear to be strong. While this band was important for this particular image and classification, the obvious signal contained in this noisy band supports the claim that high order MNF bands contain signal that may be important to an application, but the signal is difficult to use without reducing the noise. The noisy band 36 was not selected to discriminate pine species in the unfiltered image (see Table 7), but was selected in *all* of the filtered MNF images. Figure 43 compares a 9×9 filter for band 36 to the original band 36 for the full scene, showing that the variance between features in the image is noticeable once the level of noise is reduced.

Evidence supporting adaptive filters over simply truncating the MNF to reduce the noise in an image is shown in Table 8. The classification accuracies for the MNF truncated, unfiltered images are consistently less than the classification accuracies in Table 7 for

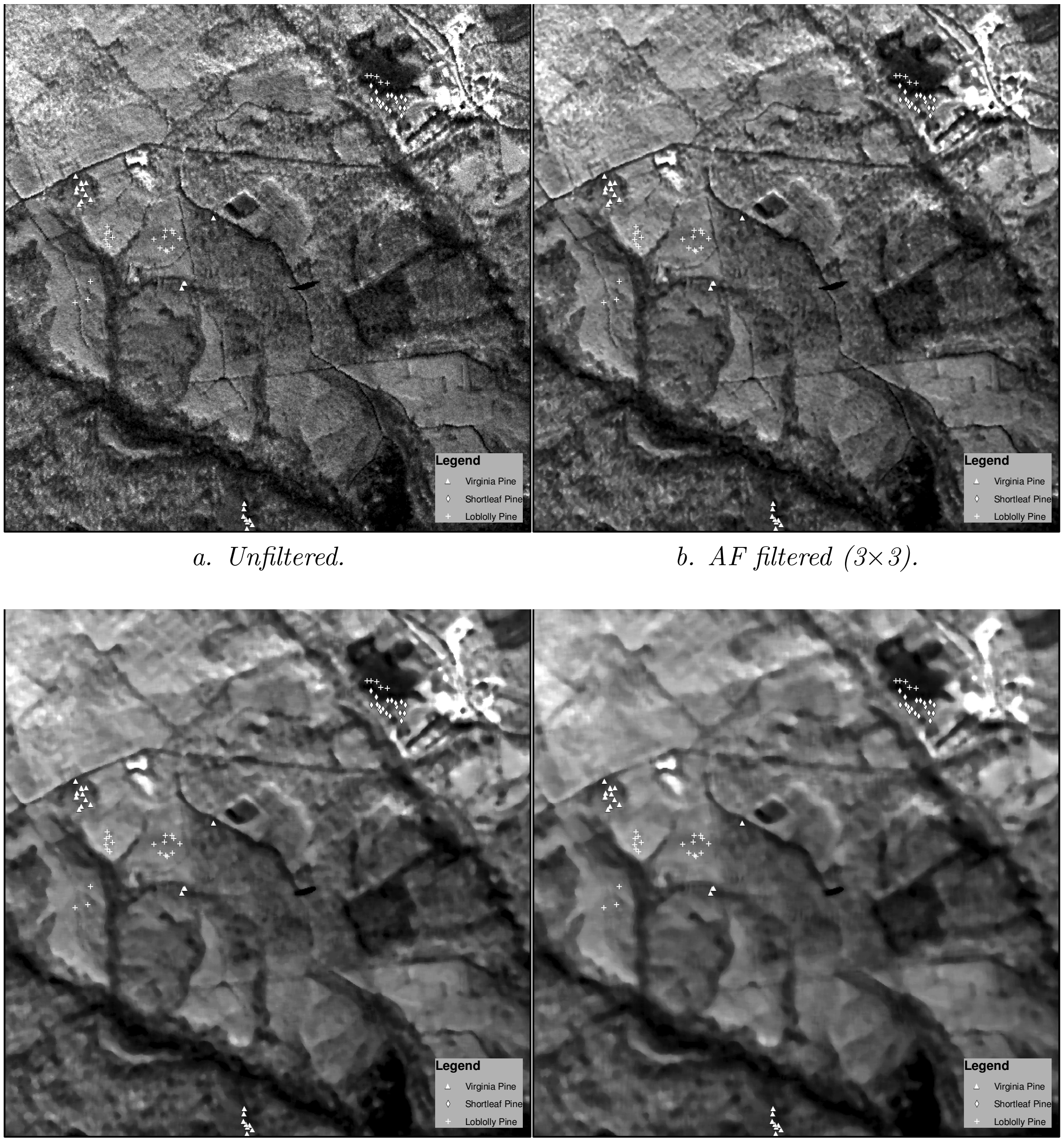


a. Unfiltered. *b. AF filtered.*

c. AFD filtered. *d. 9×9 filtered.*

Figure 40. Band 27 of ABSF (3 flights lines of AVIRIS imagery) in the original coordinate system (zoomed).

both filtering schemes when comparing classifications built using similar numbers of bands. Combining the truncation of MNF with an adaptive filtering method to entirely remove bands that have no discernible signal while filtering bands containing signal heavily degraded by noise may be another viable noise reduction technique. Since the truncated MNF will be filtered in this case, the number of bands kept can be higher to ensure that little or no signal is removed from the image. Notice in Table 8 that each adaptive filtering method using 80 MNF bands produces far better classification accuracies than keeping 40 or 60 unfiltered



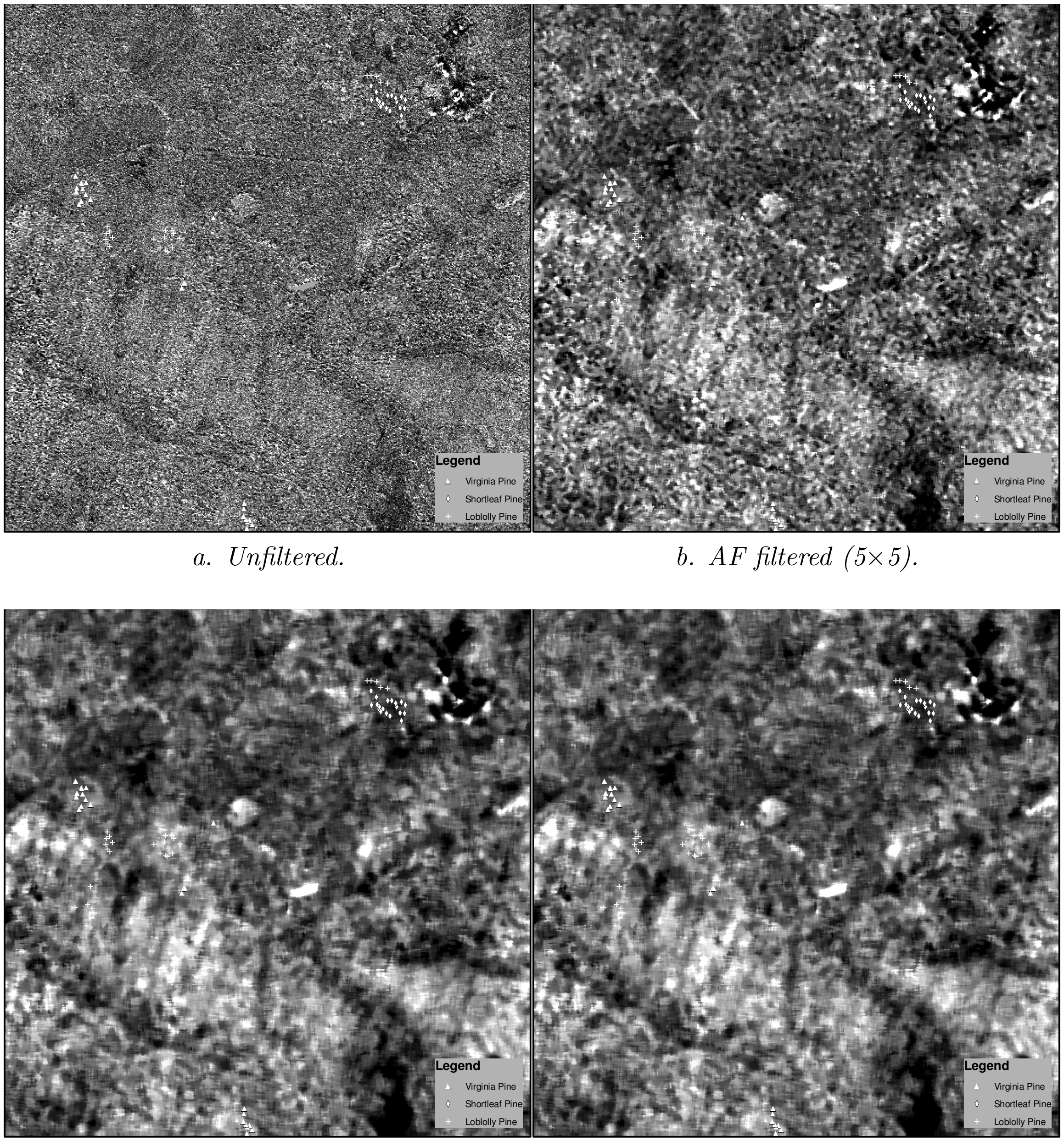
a. Unfiltered. *b. AF filtered (3×3).*

c. AFD filtered (7×7). *d. 9×9 filtered.*

Figure 41. Band 5 of the ABSF (3 flights lines of AVIRIS imagery) MNF (zoomed).

MNF bands. Also notice that accuracies are lower for the image truncated at 60 bands than the image truncated at 40 bands. Bands 40–60 have low SNRs, and including these bands without filtering them introduces far more noise than signal to the image, resulting in lower classification accuracies.

Figures 40–43 show why adaptive filtering is necessary over one uniform filter size for data transformed by the MNF. While a large filter degrades strong signal in low order bands, it can be useful to remove noise from a high order band with a weak signal. Even though the weak signal is inevitably degraded, it is practically useless when the noise is

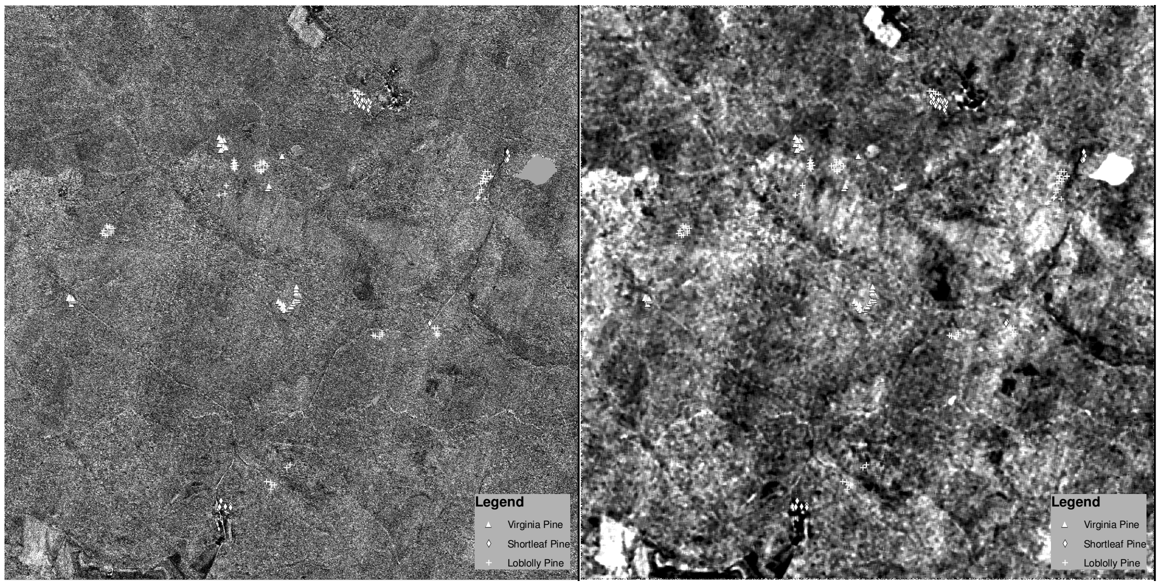


a. Unfiltered. *b. AF filtered (5×5).*

c. AFD filtered (9×9). *d. 9×9 filtered.*

Figure 42. Band 36 of the ABSF (3 flights lines of AVIRIS imagery) MNF (zoomed).

comparatively strong. Using an adaptive filter allows the strong signal in the first few bands to be preserved while the dominating noise is removed by a much larger filter in high order bands. The results presented for this particular dataset are intended to demonstrate a general technique and are not intended to indicate the ideal filter sizes and number of bins for other datasets and applications. This particular dataset and classification appeared to benefit from spatial filters with large windows, explaining why the AFD filter with a maximum filter window size of nine (very similar to a uniform 9× 9 filter) produced such accurate classification results.



a. Unfiltered. *b. 9×9 filtered.*

Figure 43. Band 36 of the ABSF (3 flights lines of AVIRIS imagery) MNF (full scene).

A reduction in texture has been demonstrated to affect classification accuracy, especially when homogeneous classes are created that are easier to classify [26]. As shown in Figures 40–43, the textural properties of an image are potentially affected by the noise filters. Figure 10 qualitatively demonstrates that the texture of the original image is reduced, particularly using large filters (AFD and 9×9). However, Figures 40 and 41 also qualitatively demonstrate that the texture reduction using AF is minimal because the MNF bands containing most of the signal (texture) are unfiltered or filtered using a small window. The reason for using an adaptive filter based on the MNF is to minimize the overall reduction in signal and texture by using large filters on bands with low SNRs and therefore low signal. Notice in Figures 42 and 43 that there is little signal or texture in the original, unfiltered image band, and the result of applying the large filter is predominantly a reduction in noise. This MNF band 36 (as well as other high numbered bands) was shown to be important in discriminating between the three pine species, indicating these classification accuracies were improved by a reduction in noise. Furthermore, the points used in this classification and validation are sparse and distributed throughout the image, and therefore are less likely to substantially benefit from a homogenizing effect that would greatly affect the ability to classify these points. Although a reduction in texture has been shown to sometimes improve classification accuracies, the evidence provided in this thesis indicates classification accuracy improvements are a result primarily of noise reduction.

Chapter 23: AF CONCLUSIONS

This work introduced an adaptive filter based on the MNF that exploits the ordering of the bands to apply median filters of different sizes. This filtering scheme greatly enhanced the MNF image for the purposes of identifying pine species, and accuracies were improved by more than 10% for certain variations of the filtering algorithm applied to AVIRIS data in the original and MNF coordinate systems. The results in this thesis are substantially more accurate than previously reported results for the same application and analysis performed on the same data in which no spatial filters were applied [91].

The AFD version of the adaptive filter produced more accurate classification results and higher estimated SNRs than the AF version, however, there are indications that the AFD degraded the signal quality significantly (while also degrading the noise), perhaps making the AF more appropriate in certain applications. Both variations led to classifications that were substantially better than classifications performed on unfiltered data. This thesis does not indicate how the number of bins should be selected. More work is needed to identify the number of bins and the maximum size of filters that should be used for certain applications and on certain imagery. Furthermore, more study is necessary to determine the relationship between SNR estimates and ideal filter size and the suitability of adaptive frequency domain filters in this context should be examined.

Section IV. CIGSCR.

Chapter 24: SOFT CLASSIFICATION INTRODUCTION

The classification of remotely sensed imagery is essential for many remote sensing applications such as natural resource management, change detection, species identification, etc. Crisp classifications assign each pixel or sample to one class in the particular classification scheme, which can be interpreted as picking the class that has the highest probability of containing the sample (when probability models are used for classification). Alternatively, soft classifications contain information on possible memberships in multiple classes, not just the most likely class. Soft or subpixel classifications are of considerable interest in the remote sensing community as this type of classification can effectively model geographic data whose natural boundaries rarely coincide with pixel boundaries. Furthermore, pixels can also contain multiple species that are commingled, leading to classification difficulty. Individual classes within the classification scheme can have overlapping electromagnetic reflectance spectra, making it difficult to discriminate between these classes. Scientists have successfully used soft classification for applications such as land cover mapping [82], vegetation mapping [58], and the classification of snow [70], to name a few. Popular methods for obtaining soft classifications of remotely sensed images include fuzzy c-means [68] and spectral unmixing [78].

Semisupervised classification has received a good deal of attention in the remote sensing community as remote sensing datasets are characterized by a large number of dimensions (hyperspectral imagery) and limited training data. While training data is expensive to obtain in any discipline, it is especially so in remote sensing as the labeling of image data typically requires extensive knowledge of the study area, multiple data sources, and/or physically visiting the study area to identify classes. Semisupervised learning can be used to supplement a labeled training set with unlabeled data to mitigate the Hughes phenomenon (overfitting of a classification when the training data is insufficient for the number of dimensions present in the dataset to be classified) [45][83].

Semisupervised classification algorithms such as the iterative guided spectral class rejection (IGSCR) algorithm ([95],[67],[71]) have the additional benefit of providing a high level of automation compared to strictly supervised classification algorithms. In remote sensing, informational class categories that make up a classification scheme are defined prior to classification and are identified by humans, whereas spectral classes or clusters have mathematical properties (such as mathematically homogeneous spectral waveforms) and are more difficult for humans to identify. For example, suppose a forest/nonforest classification is desired, and forest and nonforest are the informational class categories. Each informational class is composed of multiple spectral classes that can be used in supervised classification, and the individual spectral classes may not be spectrally similar to each other despite all being part of one informational class. Consider the wide range of tree species that could potentially make up a forest informational class in a particular image. An unsupervised technique such as clustering can identify individual classes that are mathematically homogeneous, and has the additional property of guaranteeing that all types of land cover present in a dataset are represented in the spectral classes (clusters). Both tasks are nontrivial for humans to perform when identifying spectral classes for supervised classification. Therefore semisupervised classification algorithms that involve clustering can automatically identify and label spectral classes, providing significant automation over supervised or unsupervised classification alone.

The purpose of this work is to develop a semisupervised soft clustering framework, analogous to the framework in IGSCR, that is capable of producing soft classifications of remotely sensed images. This framework will potentially affect semisupervised classification algorithms that have labeled data and involve clustering. Soft clustering retains all information regarding the proximity of data points to clusters, and will therefore directly produce a soft classification and will potentially provide better training spectral classes for a supervised decision rule. The major challenges to converting the discrete IGSCR to a fully continuous algorithm producing soft classification are in converting the underlying inherently discrete models and algorithms to suitable continuous models and algorithms while preserving the automated spectral class identification properties of IGSCR. More specifically, a hypothesis test that is fundamental to IGSCR is based on the discrete binomial probability distribution. A hypothesis test based on a new continuous probability distribution is necessary in continuous IGSCR (CIGSCR). IGSCR uses an iterative cluster refinement framework that breaks down under soft clustering, and therefore a new iterative cluster refinement method is developed for CIGSCR. Furthermore, soft clustering allows for the magnification of distances using radial functions that changes soft clusters but would have no effect on hard clusters.

Chapter 25: SEMISUPERVISED LEARNING AND CLUSTERING

Semisupervised learning occurs when unlabeled data are used in addition to labeled data to produce a classification [24]. Semisupervised learning can be more accurate than supervised learning (for a given set of labeled data) if knowledge of the underlying data distribution $p(x)$ (gained through the unlabeled data) contributes to knowledge of the conditional distribution $p(c|x)$, where c is the class label for data point x [24]. When assumptions about $p(c|x)$ are incorrect, using information about $p(x)$ can actually degrade classification accuracy [24].

An assumption commonly used in semisupervised learning is that if two particular points in a dense region are "close," their corresponding class labels should also be "close." In the context of clustering, this indicates that two points contained in the same cluster are likely to be in the same class, which is known as the "cluster assumption" [81]. Semisupervised learning methods that invoke the cluster assumption include the method proposed in [7]. Unfortunately, this assumption is sometimes not true as clusters are not necessarily composed of one class. Several clustering methods have been suggested that seek to form clusters based on both traditional clustering criteria and secondary criteria that could include a correlation between clusters and classes. Clustering methods that use additional information to influence clusters are known as *semisupervised clustering* (distinct from semisupervised learning) methods.

One method that seeks to influence the formation of clusters is clustering with constraints. In these methods, constraints are provided in the form of must-link constraints where two samples should appear in the same cluster and cannot-link constraints where two samples should not appear in the same cluster. These constraints are used with a traditional clustering method such as k-means, and the constraints can be strictly enforced algorithmically [94] or by using a modified objective function [29]. When using an objective function, there is no guarantee that all constraints will be satisfied. Basu et al. [4] suggested a method by which constraints that are informative can be selected and used in clustering, and Bilenko et al. [13] used constraints to learn a distance metric that would provide a good clustering. Halkidi et al. [40] use constraints to measure the quality of a clustering and tune Euclidean distance weight parameters to find the "best" clustering. Bouchachia and Pedryz [15] introduced a soft semisupervised clustering method with an objective function that accounts for prior information in the form of class labels. Having class labels can be viewed as a special case of having constraints as must-link and cannot-link constraints can be generated from the labeled data. Other methods that use additional information to form clusters include information bottleneck ([90], [85], [88]) and discriminative clustering [53]. These algorithms form a clustering objective function that measures distortion of the auxiliary data due to clustering.

Semisupervised learning has been used in the remote sensing community for some time to supplement limited training samples in the classification of remotely sensed images. The application of semisupervised learning to correct classification overfitting was studied in [83]. Jeon and Landgrebe used semisupervised techniques (including clustering) to perform classifications on entire images when only one class is of interest and labeled [46]. Multiple semisupervised methods based on support vector machines (SVM) have been developed for the classification of hyperspectral imagery ([17], [20]), and Gòmez-Chova et al. used clustering and SVMs to form a semisupervised classification method [37]. IGSCR also utilizes clustering in a semisupervised framework to classify remotely sensed images ([95], [67], [71]). Due to its high accuracy and automation, IGSCR is a frequently used hybrid classification method in the remote sensing community ([49], [55], [84], [96]).

Chapter 26: A SEMISUPERVISED VIEW OF IGSCR

IGSCR is a classification method that uses clustering to generate a classification model $p(c_i|x)$ where x is a multivariate sample to be classified and c_i, $i = 1, \ldots, C$, is the ith class where there are C classes in the classification scheme. IGSCR uses clustering to estimate $p(k_j|x)$ in the expression

$$p(c_i|x) = \sum_{j=1}^{K} p(c_i, k_j|x) = \sum_{j=1}^{K} p(c_i|k_j, x)p(k_j|x), \tag{1}$$

where k_j, $j = 1, \ldots, K$, is the jth cluster out of K total clusters. IGSCR also uses the clusters to train a decision rule using Bayes' theorem [36]

$$p(k_j|x) = \frac{p(x|k_j)p(k_j)}{\sum_{i=1}^{K} p(x|k_i)p(k_i)}. \tag{2}$$

The prior probabilities of the clusters $p(k_j)$ are assumed to be equal.

Clustering is performed using a discrete clustering method such as k-means that minimizes the objective function

$$J(\rho) = \sum_{i=1}^{n}\sum_{j=1}^{K} w_{ij}\rho_{ij} \tag{3}$$

subject to

$$\sum_{j=1}^{K} w_{ij} = 1$$

where $w_{ij} \in \{0, 1\}$ is the value in the ith row and jth column of the partition matrix $W \in \Re^{n\times K}$, $U^{(j)} \in \Re^B$ is the prototype for the jth cluster k_j, $x^{(i)} \in \Re^B$ is the ith data point, and $\rho_{ij} = ||x^{(i)} - U^{(j)}||_2^2$. The clusters $k_1, \ldots, k_K$ form a partition of $\{x^{(i)}\}_{i=1}^{n}$. The algorithm for k-means requires K initial cluster prototypes and iteratively assigns each sample to the closest cluster using

$$w_{ij} = \begin{cases} 1, & \text{if } j = \operatorname*{argmin}_{1\le j\le K} \rho_{ij}, \\ 0, & \text{otherwise,} \end{cases}$$

followed by the cluster prototype (mean) recalculation

$$U^{(j)} = \sum_{i=1}^{n}(w_{ij}x^{(i)}) \Big/ \sum_{i=1}^{n} w_{ij}$$

once W has been calculated [34]. This process, guaranteed to terminate in a finite number of iterations, continues until no further improvement is possible, terminating at a local minimum point of (3).

IGSCR uses labeled data in a semisupervised clustering framework to locate clusters that map to classes in a given classification scheme. IGSCR requires a labeled set of training data comprised of individual samples within the image to be classified and corresponding class labels. Rather than using the labeled data to train a decision rule directly, the entire image is clustered, thereby capturing the inherent structure of all the data and not just the labeled samples. The clusters represent spectral classes, and in remote sensing, each spectral class ideally maps to exactly one class in the final classification scheme. Once clusters are generated, each cluster must be mapped to one class or rejected as impure. While theoretically each cluster should contain samples belonging to only one informational class, in practice clusters (spectral classes) that contain predominantly samples of one class can contain a few samples from other classes because of inherent errors. However, if a cluster contains too many samples from different classes, the cluster itself is considered confused and should not be labeled with one class. Impure clusters are rejected and can be further refined in the iterative part of the algorithm.

The test for cluster purity is performed using the labeled training set. IGSCR produces a hard classification and uses a discrete clustering method where each sample is assigned to exactly one cluster. Let $V_{c,j}$ be the binomial random variable denoting the number of labeled samples assigned to the jth cluster that are labeled with a particular cth class. Let p be the user-supplied cluster homogeneity threshold ($p = .9$ would indicate a cluster is 90% pure with respect to the majority class), and let α be the user-supplied acceptable one-sided Type-I error for a statistical hypothesis test. Then if c is the majority class represented in the jth cluster, the jth cluster is rejected if $P(Z < \hat{z}) < 1 - \alpha$ where Z is a standard normal random variable, m is the number of labeled samples in the jth cluster, and

$$\hat{z} = \frac{v_{c,j} - mp}{\sqrt{mp(1-p)}}. \tag{4}$$

(Typically a continuity correction of 0.5 is added in the numerator of (4).)

If a cluster is rejected, the samples making up that cluster can be reclustered in subsequent iterations. All samples belonging to pure clusters are removed from the image being clustered, resulting in only samples belonging to impure clusters being reclustered. Once more clusters are generated, those clusters are evaluated for purity, removed from the image, and clustering is performed again until termination criteria are met. All samples can belong to pure clusters, leaving no remaining samples to be clustered, no pure clusters could be found in the previous iteration, meaning that the clustering would continue to be performed on the same data, resulting in the same impure clusters (assuming deterministic cluster seeding), or a set number of iterations can be reached, resulting in termination of the iteration. Note that deterministic seeding ensures that the iteration will terminate, even without specifying a maximum number of iterations.

Once the iterative clustering is complete, one or more classifications is performed. The first classification is called the iterative stacked (IS) classification because it is the result of combining or "stacking" all cluster assignments over all iterations (each sample will be assigned to at most one accepted cluster). Assume that all samples not assigned to an accepted cluster are combined to form one cluster k_{K+1}, and the class assignment for that cluster is "unclassified" or c_{C+1}. Then the IS assignment for a pixel using (1) is

$$\mathrm{IS}(x) = \operatorname*{argmax}_{1 \le i \le C+1} p(c_i|x) = \operatorname*{argmax}_{1 \le i \le C+1} \sum_{j=1}^{K+1} p(c_i|k_j, x) p(k_j|x),$$

where

$$p(c_i|k_j, x) = \begin{cases} 1, & \text{if } k_j \text{ is labeled } c_i, \\ 0, & \text{otherwise,} \end{cases}$$

and

$$p(k_j|x) = \begin{cases} 1, & \text{if } x \in k_j, \\ 0, & \text{otherwise,} \end{cases}$$

since cluster assignments are discrete.

The second possible classification, the decision rule (DR) classification, uses the pure clusters to form a decision rule. Recall in (2) that

$$p(k_j|x) = \frac{p(x|k_j)}{\sum_{i=1}^{K} p(x|k_i)}$$

when all the $p(k_j)$ are equal. Traditionally, the maximum likelihood decision rule, assuming a multivariate normal distribution

$$p(x|k_j) = 2\pi^{-B/2}|\Sigma_j|^{-1/2}e^{-\frac{1}{2}(x-U^{(j)})^T\Sigma_j^{-1}(x-U^{(j)})},$$

is used where Σ_j is the covariance matrix of the jth cluster [77]. Since IGSCR produces hard classifications, the full probability need not be calculated as determining only the cluster associated with the maximum probability is necessary. The DR classification function is

$$\text{DR}(x) = \operatorname*{argmax}_{1\le i\le C} p(c_i|x) = \operatorname*{argmax}_{1\le i\le C} \sum_{j=1}^{K} p(c_i|k_j, x)p(k_j|x), \tag{5}$$

where

$$p(k_j|x) = \begin{cases} 1, & \text{if } j = \operatorname*{argmax}_{1\le j\le K}\big(-\ln|\Sigma_j| - (x-U^{(j)})^T\Sigma_j^{-1}(x-U^{(j)})\big), \\ 0, & \text{otherwise.} \end{cases}$$

A final classification, the iterative stacked plus (IS+) classification, combines the DR and IS classifications. If a sample is labeled as unclassified in the IS classification, the DR class value is used for the IS+ classification, otherwise the IS class value is used for that particular sample. The IS+ classification function is

$$\text{IS+}(x) = \begin{cases} \text{IS}(x), & \text{if } x \notin k_{K+1}, \\ \text{DR}(x), & \text{otherwise.} \end{cases}$$

Chapter 27: CIGSCR

Continuous IGSCR (CIGSCR) uses a similar semisupervised clustering framework to the one established in IGSCR to produce a soft or probabilistic classification instead of a hard classification, and uses continuous algorithms and models instead of discrete algorithms and models. Recall in (1) that $p(c_i|k_j,x)$ and $p(k_j|x)$ are either 0 or 1 (discrete) in practice in IGSCR. $p(c_i|k_j,x)$ is necessarily discrete because while several clusters can comprise one class, only one class (theoretically) can label the members of a particular cluster, but there are no similar restrictions on $p(k_j|x)$. In fact, the clustering algorithm and the maximum likelihood decision rule indicate positive probabilities that a sample is associated with each cluster, but IGSCR makes an assignment only to the cluster with the highest probability.

Consider a soft clustering algorithm that minimizes the objective function [10]

$$\begin{aligned} J(\rho) = \sum_{i=1}^{n}\sum_{j=1}^{K} w_{ij}^{p}\rho_{ij} \quad &\text{subject to} \\ \sum_{j=1}^{K} w_{ij} = 1 &\text{ for each } i \end{aligned} \tag{6}$$

where $w_{ij} \in (0,1)$ is the value in the ith row and jth column of the weight matrix $W \in \Re^{n\times K}$ (analogous to the partition matrix W in (3)), $U^{(j)} \in \Re^{B}$ is the jth cluster prototype, $p > 1$, and $\rho_{ij} = \rho(x^{(i)}, U^{(j)}) = ||x^{(i)} - U^{(j)}||_2^2$ is the Euclidean distance squared. The algorithm that minimizes this objective function is similar to that of k-means in that it first calculates

$$w_{ij} = \frac{(1/\rho_{ij})^{1/(p-1)}}{\displaystyle\sum_{k=1}^{K}(1/\rho_{ik})^{1/(p-1)}}$$

for all i and j followed by calculating updated cluster prototypes

$$U^{(j)} = \sum_{i=1}^{n} w_{ij}^{p} x^{(i)} \Big/ \sum_{i=1}^{n} w_{ij}^{p}.$$

This iteration (recalculation of the weights followed by recalculation of cluster prototypes, following by recalculation of the weights, etc.) is guaranteed to converge (with these definitions of ρ_{ij}, $U^{(j)}$, and w_{ij}) for $p > 1$ [11].

With a continuous alternative to the discrete hypothesis test and a continuous alternative to the IGSCR iterative cluster refinement that follows in Chapters 28 and 29, the classification function for IS classification is

$$\mathrm{IS}(x) = p(c_i|x) = \sum_{j=1}^{K} p(c_i|k_j,x)p(k_j|x), \tag{7}$$

where $p(k_j|x)$ is estimated using w_{ij} and $p(c_i|k_j,x)$ does not change from IGSCR. The classification function for the DR classification is

$$\mathrm{DR}(x) = p(c_i|x) = \sum_{j=1}^{K} p(c_i|k_j,x)p(k_j|x)$$

$$= \frac{\displaystyle\sum_{j=1}^{K} p(c_i|k_j,x)\left[\frac{2e^{-\frac{1}{2}(x-U^{(j)})^T\Sigma_j^{-1}(x-U^{(j)})}}{\pi^{B/2}|\Sigma_j|^{1/2}}\right]}{\displaystyle\sum_{l=1}^{K}\left[\frac{2e^{-\frac{1}{2}(x-U^{(l)})^T\Sigma_l^{-1}(x-U^{(l)})}}{\pi^{B/2}|\Sigma_l|^{1/2}}\right]}. \tag{8}$$

An analog for the IS+ classification is unnecessary in CIGSCR as all samples will be part of pure clusters and will be classified.

Chapter 28: ASSOCIATION SIGNIFICANCE TEST

A key component in the IGSCR semisupervised clustering framework is the homogeneity test used to determine if a cluster contains a statistically significant proportion of one class. This test provides a basis for rejecting a cluster for further refinement, the second phase of the semisupervised clustering.

A cluster might be composed of more than one class because the cluster itself is in fact composed of more than one cluster. A cluster might also contain more than one class because the initial clusters were determined in such a way as to prevent a cluster from moving toward a particular class. It would be useful to determine which clusters are not spectrally pure (contain more than one class with high probability) so that the cluster can be further refined, and if no refinement is possible (any number of iteration ending criteria are met), the cluster should not be used in the classification model. Statistical hypothesis tests provide a mechanism for determining class purity once an appropriate statistical model is selected for the data.

In hard IGSCR with hard clustering, the notion of a pure cluster is clear. Each sample will belong to one and only one cluster. A cluster can be 100% homogeneous when all labeled samples contained within that cluster belong to only one class. Although this is possible, it is unlikely that one cluster contains only one class because of inherent error in the labeling process and because two different informational class categories can contain spectrally similar samples. Once a homogeneity level is determined, a rigorous hypothesis test can be applied to select clusters that contain a certain percentage of one class, with that percentage unlikely to be observed in a particular cluster randomly.

Using soft clusters introduces complications to assessing and determining cluster purity. The first question might be whether a soft cluster can be spectrally pure, because being soft might indicate that clusters are naturally comprised of multiple classes. However, just as the goal in IGSCR is to determine clusters that are representative of just one predominant class, that goal holds in CIGSCR with soft clusters. Soft clusters are composed of different portions of each sample or pixel within an image, meaning that each sample has a positive probability of being in different individual classes or clusters. When samples labeled with different classes have a positive probability of belonging to the same cluster, that does not indicate that the cluster really contains two different classes, but rather perhaps that while the pixels have strong associations with different classes, there is also a positive (although possibly small) probability that each pixel actually belongs to or partially belongs to the majority class within the cluster. Both cases (the cluster is confused or the cluster is not confused but the pixels labeled with different classes still have small associations with the same class) are possible in soft clustering. The appropriate test for soft clusters is not which pixels "belong" to a particular cluster (they all "belong" to some degree), rather how strongly pixels from different classes belong to a particular cluster. If pixels from only one class have strong associations with a cluster when compared to pixels labeled with other classes, then the cluster should be labeled with that most strongly associated class. In this manner, each pixel/sample is associated by varying degrees with multiple spectrally pure clusters that are mapped to individual classes, ultimately producing a soft classification output when each sample is then mapped to different individual classes with varying probabilities.

28.1 Distribution

Developing a hypothesis test to assess purity of clusters requires a random variable and knowledge of the distribution of that random variable. In IGSCR, a cluster can be considered pure and labeled with a class if the number of labeled samples belonging to the class is high compared to the number of labeled samples not belonging to the class. The random variable of interest, $V_{c,j} = \sum_{i \in I_j} V_{ic}$, is the count of the number of labeled samples belonging to the cth class for a particular jth cluster where i is the pixel index, I_j is the index set of labeled pixels in the jth cluster, and V_{ic} is the Bernoulli random variable corresponding to the ith pixel being associated with the cth class. A hypothesis test can be developed using the binomial distribution, or the less computationally intensive normal distribution, which approximates the binomial distribution well when the number of labeled samples is large.

In CIGSCR, the random variable and distribution are more complicated as there are class memberships (either 0 or 1) and cluster memberships (between 0 and 1). Building a test on only the class memberships is not useful as each labeled sample will have some positive probability of belonging to a particular cluster, making the results of the test the same for each cluster unless memberships are also considered. In this case, the association of a sample to a particular class (the majority class, for example) is still a Bernoulli trial. Each pixel also has a weight vector, $w_{i\cdot}$, indicating the probability of membership to each cluster. The random variable of interest is the sum of the memberships for the cth class and weights to the jth cluster,

$$Y_{c,j} = V_{1c}W_{1j} + V_{2c}W_{2j} + \cdots + V_{nc}W_{nj},$$

where n is the total number of labeled samples. The labels of the classified pixels are independent of cluster assignment, making an assumption that V_{ic} and W_{ij} are independent reasonable. Furthermore, the training samples are labeled prior to clustering, making the random variable of interest

$$Y_{c,j}|(V_{1c}, V_{2c}, \ldots, V_{nc}) = \sum_{i=1}^{n} W_{ij}\delta_{\phi(i),c},$$

where $\phi(i)$ is the label of the ith pixel, and

$$\delta_{\phi(i),c} = \begin{cases} 0 \text{ if } \phi(i) \neq c, \\ 1 \text{ if } \phi(i) = c, \end{cases}$$

is the Kronecker delta. The probability density function (pdf) of $Y_{c,j}|(V_{ic}, i = 1, \ldots, n) = \sum_{i=1}^{n} W_{ij}\delta_{\phi(i),c}$ is the pdf of a sum of individual cluster weights.

Figures 44 and 45 contain experimental frequency histograms of weights w_{ij} for two clusters ($K = 2$) of a satellite image. The distribution of the cluster weights appears to be multimodal, which is consistent with the data having multiple inherent classes, indicating that W_{ij}, $i = 1, \ldots, n$, $j = 1, \ldots, K$ would not be identically distributed. A closed form distribution is not readily available for W_{ij}, but a closed form distribution, or at least a reasonable approximate closed form distribution, for $W_{+j} = \sum_{i=1}^{n} W_{ij}$ exists.

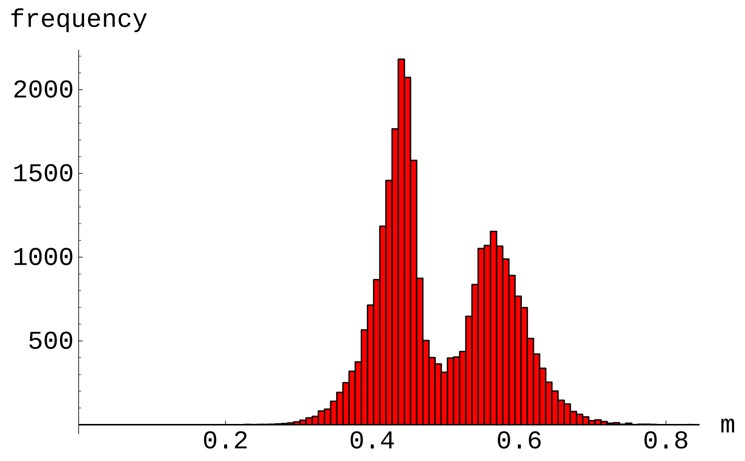


Figure 44. Histogram of cluster weights in one cluster, K=2.

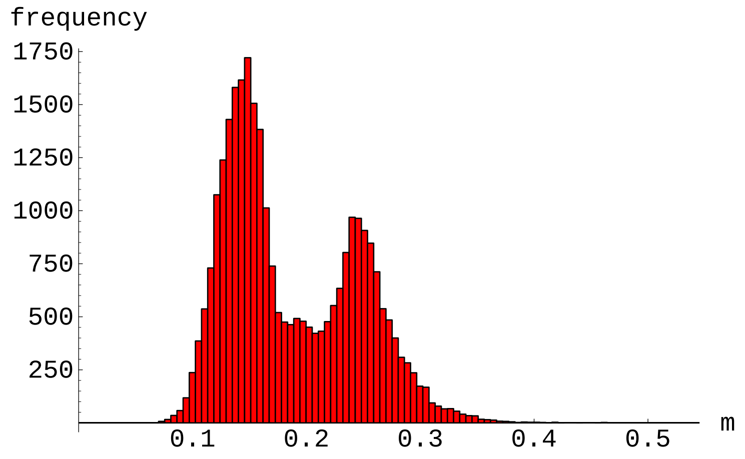


Figure 45. Histogram of cluster weights in one cluster, K=5.

28.2 Normal Approximation to $Y_{c,j}$

Suppose an image x contains n pixels $x^{(i)} \in \Re^B$, $i = 1, \ldots, n$. For K fixed cluster centers $U^{(k)} \in \Re^B$, $k = 1, \ldots, K$, the assigned weight of the ith pixel to the jth cluster is

$$w_{ij} = \frac{1/||x^{(i)} - U^{(j)}||_2^2}{1\Big/\sum_{k=1}^{K} ||x^{(i)} - U^{(k)}||_2^2},$$

which is the inverse of the distance squared over the sum of the inverse squared distances. (Such inverse distance weights are widely used, e.g., by Shepard's algorithm for sparse data interpolation.) Note this is the specific case in the soft clustering algorithm described above when $p = 2$. In this case where a remotely sensed image is to be clustered, it is reasonable to assume that $x^{(i)}$, $i = 1, \ldots, n$ are generated from a finite number of multivariate normal distributions. The act of clustering assumes that the data are generated from a finite number of distributions, and remotely sensed earth data are assumed to be generated from normal distributions. The following proof demonstrates that under these assumptions (pixels are generated from a finite number of normal distributions), the Lindeberg condition is satisfied and therefore the central limit theorem applies to the sum of a sequence of cluster weight random variables $\sum_{i=1}^{n} W_{ij}$. Let $q = \psi(i)$ denote the distribution from which $X^{(i)}$ was sampled.

Theorem: Let $X^{(i)}$, $i = 1, 2, \ldots$, be B-dimensional random vectors having one of Q distinct multivariate normal distributions. For $i = 1, 2, \ldots$ and $j = 1, \ldots, K$ define the random variables

$$W_{ij} = W_j(X^{(i)}) = \frac{1/||X^{(i)} - U^{(j)}||_2^2}{\sum_{k=1}^{K} 1/||X^{(i)} - U^{(k)}||_2^2},$$

where K is the number of clusters and $U^{(k)} \in \Re^B$ is the kth cluster center (and is considered fixed for weight calculation). Then for any $j = 1, \ldots, K$,

$$P\left\{\frac{1}{B_{nj}}\sum_{i=1}^{n}(W_{ij} - a_{ij}) < x\right\} \to \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{z^2}{2}}dz$$

as $n \to \infty$, where $a_{ij} = \mathrm{E}[W_{ij}]$, $b_{ij}^2 = \mathrm{Var}[W_{ij}]$, and $B_{nj}^2 = \sum_{i=1}^{n} b_{ij}^2$.

Proof. W_{ij} is a bounded ($0 \le W_{ij} \le 1$) measurable function of a normal random variable, and is therefore a random variable with finite mean and variance. Fix j for the remainder of the proof, and let $q = \psi(i)$ denote which of the Q distributions $X^{(i)}$ is from. In order to prove

$$P\left\{\frac{1}{B_{nj}}\sum_{i=1}^{n}(W_{ij} - a_{ij}) < x\right\} \to \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{z^2}{2}}dz,$$

it is sufficient to verify the Lindeberg condition [36]:

$$\lim_{n\to\infty}\frac{1}{B_{nj}^2}\sum_{i=1}^{n}\int_{|x-a_{ij}|>\tau B_{nj}}(x - a_{ij})^2 dF_{\psi(i),j}(x) = 0,$$

for any constant $\tau > 0$ where $F_{\psi(i),j}(x)$ is the cumulative distribution function for W_{ij}.

For each q, $1 \le q \le Q$, define $I_q = \psi^{-1}(q) = \{i \mid \psi(i) = q, 1 \le i \le n\}$, $n_q = |I_q|$, and for $i \in I_q$ let $\mathrm{E}[W_{ij}] = a_{ij} = \alpha_{qj}$ and $\mathrm{Var}[W_{ij}] = b_{ij}^2 = \beta_{qj}^2$. Now considering only the independent and identically distributed random variables W_{ij}, $i \in I_q$, the Lindeberg condition holds:

$$\begin{aligned}&\lim_{n_q\to\infty}\frac{1}{n_q\beta_{qj}^2}\sum_{i\in I_q}\int_{|x-\alpha_{qj}|>\tau\sqrt{n_q}\beta_{qj}}(x - \alpha_{qj})^2 dF_{qj}(x)\\ &= \lim_{n_q\to\infty}\frac{1}{\beta_{qj}^2}\int_{|x-\alpha_{qj}|>\tau\sqrt{n_q}\beta_{qj}}(x - \alpha_{qj})^2 dF_{qj}(x) = 0.\end{aligned}$$

Since β_{qj} is positive and finite, and the integral is finite, the limit of the integral is zero as $\sqrt{n_q}\beta_{qj} \to \infty$.

$W_{ij}, i = 1, 2, \ldots,$ are random variables from Q iid distributions, $F_{qj}, q = 1, \ldots, Q$, where the mean of the qth distribution is α_{qj}, the variance is β_{qj}^2, and the number of random variables from that distribution is n_q, where $\sum_{q=1}^{Q} n_q = n$. As $n \to \infty$ there is at least one q for which $n_q \to \infty$. For this sequence of independent random variables from Q distributions, the Lindeberg condition is

$$\begin{aligned}
&\lim_{n\to\infty} \frac{1}{B_{nj}^2} \sum_{i=1}^{n} \int_{|x-a_{ij}|>\tau B_{nj}} (x - a_{ij})^2 dF_{\psi(i),j}(x) \\
&= \lim_{n\to\infty} \frac{1}{\displaystyle\sum_{k=1}^{Q} n_k \beta_{kj}^2} \sum_{q=1}^{Q} n_q \\
&\qquad \cdot \int_{|x-\alpha_{qj}|>\tau B_{nj}} (x - \alpha_{qj})^2 dF_{qj}(x) \\
&= \lim_{n\to\infty} \sum_{q=1}^{Q} \frac{n_q}{\sum_{k=1}^{Q} n_k \beta_{kj}^2} \\
&\qquad \cdot \int_{|x-\alpha_{qj}|>\tau B_{nj}} (x - \alpha_{qj})^2 dF_{qj}(x) \\
&\leq \lim_{n\to\infty} \sum_{q=1}^{Q} \frac{1}{\beta_{qj}^2} \int_{|x-\alpha_{qj}|>\tau B_{nj}} (x - \alpha_{qj})^2 dF_{qj}(x) = 0.
\end{aligned}$$

Since each variance β_{qj}^2 is positive and finite, and $B_{nj} = \sqrt{n_1\beta_{1j}^2 + \cdots, +n_Q\beta_{Qj}^2} \to \infty$ as at least one $n_q \to \infty$, each integral converges to zero as $n \to \infty$, and the Lindeberg condition is verified. Q.E.D.

Remark: The assumption that the $X^{(i)}$, $i = 1, 2, \ldots$, are generated from a finite number of normal distributions is stronger than necessary. This proof holds if $X^{(i)}$, $i = 1, 2, \ldots$, are generated from a finite number of arbitrary distributions.

Experimental results match this theoretical result, as illustrated by one experiment in Figure 46.

28.3 Association Significance Test

The hypothesis test used in IGSCR to assess the significance of a cluster association to a class is based on the normal approximation to the binomial distribution (4). The null hypothesis is that the true probability of a pixel belonging to the majority class (for the cluster of interest) is less than p_0, a user supplied value. If $P(Z > \hat{z}) < \alpha$, where α is the user provided Type-I error, then the null hypothesis is rejected. The null hypothesis corresponds to the case when the cluster is impure, and rejecting the null hypothesis equates with labeling the cluster pure; if the null hypothesis is *not* rejected, the cluster is impure and the cluster is "rejected."

The hypothesis test for pure clusters in CIGSCR is different as the Bernoulli trials are fixed and testing the probability p of a success is no longer relevant. A pure soft cluster

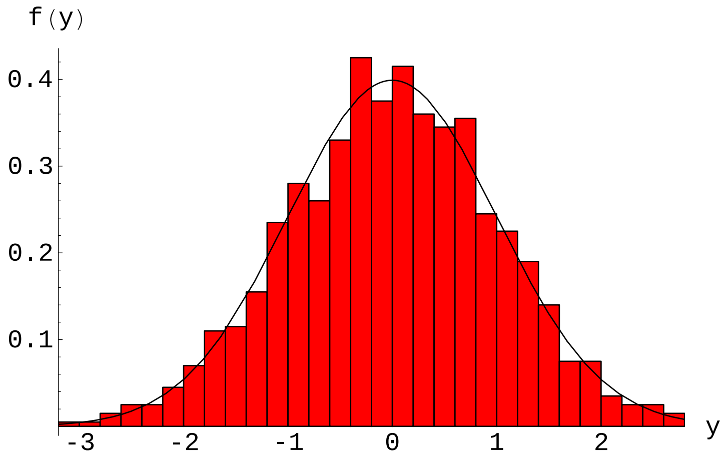


Figure 46. Pdf of Y (with sample mean subtracted and divided by the standard deviation) compared to a standard normal distribution.

should have large weights for the majority class and comparatively small weights for other classes. One possible hypothesis test compares the average weight for one particular cth class with the overall average weight for all classes in the jth cluster. Starting with the normal approximation for the sum of the cluster weights, the standard normal test statistic would be

$$\hat{z} = \frac{\sum_{i \in J_c} \left(w_{ij} - \mathrm{E}[W_{ij}] \right)}{\sqrt{\sum_{i \in J_c} \mathrm{Var}[W_{ij}]}},$$

where J_c is the index set of pixels prelabeled with the cth class. $\mathrm{E}[W_{ij}]$ and $\mathrm{Var}[W_{ij}]$ are unknown, but can be reasonably approximated using the sample mean

$$\overline{w}_j = \frac{1}{n} \sum_{i=1}^{n} w_{ij}$$

and sample standard deviation

$$S_{\overline{w}_j} = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (w_{ij} - \overline{w}_j)^2}.$$

The Wald statistic is then

$$\hat{z} = \frac{\sqrt{n_c}(\overline{w}_{c,j} - \overline{w}_j)}{S_{\overline{w}_j}}, \tag{9}$$

where $n_c = |J_c|$ and

$$\overline{w}_{c,j} = \frac{1}{n_c} \sum_{i \in J_c} w_{ij}.$$

Since $\hat{z}$ is generated (approximately) by the standard normal distribution, a hypothesis test can be formed where the null hypothesis is that the average cluster weights corresponding to the cth class *are not* significantly different from the average cluster weights corresponding to all classes, and the alternate hypothesis is that the average cluster weights corresponding to the cth class *are* significantly different from the average cluster weights corresponding to all classes. Again, since class memberships are known a priori and all pixels have some positive membership with all clusters, testing for class memberships is not meaningful, but testing for significantly different cluster weights is meaningful. If $P(Z > \hat{z}) < \alpha$, the probability of observing the difference in the average cluster weights associated with c and the average cluster weights associated with all classes in the jth cluster is significant, and the null hypothesis is rejected. If the null hypothesis is *not* rejected, the cluster itself is rejected as impure, and further refinement is necessary.

One potential issue with the above test is that the sample mean and standard deviation calculations assume the sample is identically distributed, which is specifically *not* the assumption in this case. A better hypothesis test acknowledges that the data are not identically distributed, but are generated from a finite number of distributions. Since the number of distributions and the distributions are unknown, the number of classes and the individual class labels, which are assumed to correspond to inherent structure of the data, are used to approximate the true mean and variance of multiple clusters. Precisely, assume that all labeled pixel indices i with distribution index $\psi(i) = q$ correspond to the same class label $\phi(i) = c$. If $i \in \psi^{-1}(q)$, then $i \in \phi^{-1}(c)$, but $i \in \phi^{-1}(c)$ does not imply $i \in \psi^{-1}(q)$ (more than one distribution can map to one class), and $J_c = \phi^{-1}(c) = \{i \mid \phi(i) = c, 1 \leq i \leq n\}$. The above hypothesis test requires modification to use class information. In the previous test,

$$\sum_{i \in J_c} w_{ij} = \sum_{i=1}^{n} w_{ij}\delta_{\phi(i),c},$$

$$\hat{z} = \frac{\displaystyle\sum_{i=1}^{n} \big(w_{ij}\delta_{\phi(i),c} - \mathrm{E}[W_{ij}\delta_{\phi(i),c}]\big)}{\sqrt{\displaystyle\sum_{i=1}^{n} \mathrm{Var}[W_{ij}\delta_{\phi(i),c}]}},$$

$$\begin{aligned}
&\sum_{i=1}^{n}(w_{ij}\delta_{\phi(i),c} - \mathrm{E}[W_{ij}\delta_{\phi(i),c}]) \\
&= \sum_{i=1}^{n}(w_{ij}\delta_{\phi(i),c} - a_{ij}\delta_{\phi(i),c}) \\
&= \sum_{i=1}^{n}(w_{ij}\delta_{\phi(i),c} - \alpha_{qj}\delta_{\phi(i),c}),
\end{aligned}$$

recalling that $\mathrm{E}[W_{ij}] = a_{ij} = \alpha_{qj}$ for $i \in I_q$. Assume when $\phi(i) = c$, and distribution index $q = \psi(i)$ corresponds to $c = \phi(i)$, then α_{qj} can be approximated by γ_{cj}, the mean of class $c = \phi(i)$. Ideally α_{qj} should be approximated directly, but there is no way to know $\psi^{-1}(q)$, so essentially $\psi^{-1}(q) \subset \phi^{-1}(c)$ is being approximated by $\phi^{-1}(c)$. Unfortunately, using the sample mean of the cth class and the jth cluster to approximate γ_{cj} and therefore α_{qj} breaks down because the sample mean of the cth class and the jth cluster is both the random variable on the left side and the approximation of the expected value on the right side of the minus sign. This is illustrated below. Approximating γ_{cj} (and α_{qj}) with the sample mean for the cth class,

$$\gamma_{cj} \approx \overline{w}_{c,j} = \frac{\displaystyle\sum_{k=1}^{n} w_{kj}\delta_{\phi(k),c}}{\displaystyle\sum_{k=1}^{n} \delta_{\phi(k),c}},$$

the numerator of the test statistic $\hat{z}$ becomes

$$\begin{aligned}
&\sum_{i=1}^{n}\big(w_{ij}\delta_{\phi(i),c} - \overline{w}_{c,j}\delta_{\phi(i),c}\big) \\
&= \sum_{i=1}^{n} w_{ij}\delta_{\phi(i),c} - \frac{\displaystyle\sum_{k=1}^{n} w_{kj}\delta_{\phi(k),c}}{\displaystyle\sum_{k=1}^{n} \delta_{\phi(k),c}}\sum_{i=1}^{n}\delta_{\phi(i),c} \\
&= \sum_{i=1}^{n} w_{ij}\delta_{\phi(i),c} - \sum_{k=1}^{n} w_{kj}\delta_{\phi(k),c} = 0.
\end{aligned}$$

Thus this test statistic does not work because the value being tested is the same as the estimated mean for the cth class when using the Kronecker delta instead of Bernoulli random variables. Recall that $Y_{c,j} = \sum_{i=1}^{n} V_{ic}W_{ij}$, where V_{ic}, $i = 1, \ldots, n$ are known prior to classification/clustering. Consider now the test statistic

$$\hat{z} = \frac{y_{c,j} - \mathrm{E}[Y_{c,j}]}{\sqrt{\mathrm{Var}[Y_{c,j}]}}.$$

Fixing j and c, and recalling that $n_q = |I_q|$, the number of indices i for which $X^{(i)}$ has the qth distribution,

$$\begin{aligned}
\mathrm{E}[Y_{c,j}] &= \mathrm{E}\left[\sum_{i=1}^{n} W_{ij}V_{ic}\right] = \sum_{i=1}^{n}\mathrm{E}[W_{ij}V_{ic}] \\
&= \sum_{i=1}^{n}\mathrm{E}[W_{ij}]\mathrm{E}[V_{ic}] = \sum_{q=1}^{Q} n_q\alpha_{qj}p_c = p_c\sum_{q=1}^{Q} n_q\alpha_{qj},
\end{aligned}$$

where p_c is the probability that $V_{ic} = 1$. Assuming all the pixels are independent and recalling that $\mathrm{Var}[W_{ij}] = b_{ij}^2 = \beta_{qj}^2$ where $i \in I_q$,

$$\begin{aligned}
\mathrm{Var}[Y_{c,j}] &= \mathrm{Var}\left[\sum_{i=1}^{n} W_{ij}V_{ic}\right] = \sum_{i=1}^{n} \mathrm{Var}[W_{ij}V_{ic}] \\
&= \sum_{i=1}^{n} \big(\mathrm{E}[W_{ij}^2 V_{ic}^2] - \mathrm{E}[W_{ij}V_{ic}]^2\big) \\
&= \sum_{i=1}^{n} (p_c \mathrm{E}[W_{ij}^2] - p_c^2 a_{ij}^2) \\
&= \sum_{i=1}^{n} (p_c(b_{ij}^2 + a_{ij}^2) - p_c^2 a_{ij}^2) \\
&= \sum_{q=1}^{Q} n_q \big(p_c(\beta_{qj}^2 + \alpha_{qj}^2) - p_c^2 \alpha_{qj}^2\big) \\
&= p_c \sum_{q=1}^{Q} n_q (\beta_{qj}^2 + (1-p_c)\alpha_{qj}^2).
\end{aligned}$$

In the above formula, p_c would be approximated by its maximum likelihood estimate $n_c/n = |J_c|/n$. In order to estimate α_{qj}, assume that the qth distribution corresponds to the cth class, $\psi^{-1}(q) \subset \phi^{-1}(c)$, and

$$\alpha_{qj} \approx \overline{w}_{c,j} = \frac{1}{n_c} \sum_{i \in J_c} w_{ij}, \quad c = 1, \ldots, C,$$

where C is the number of classes. Then

$$\begin{aligned}
\mathrm{E}[Y_{c,j}] &= p_c \sum_{q=1}^{Q} n_q \alpha_{qj} \approx p_c \sum_{d=1}^{C} n_d \cdot \frac{1}{n_d} \sum_{i \in J_d} w_{ij} \\
&= \frac{n_c}{n} \sum_{i=1}^{n} w_{ij} = n_c \overline{w}_j,
\end{aligned}$$

and

$$\begin{aligned}
\mathrm{Var}[Y_{c,j}] &= p_c \sum_{q=1}^{Q} n_q (\beta_{qj}^2 + (1-p_c)\alpha_{qj}^2) \\
&\approx p_c \sum_{d=1}^{C} n_d (S_{\overline{w}_{d,j}}^2 + (1-p_c)\overline{w}_{d,j}^2),
\end{aligned}$$

where

$$S_{\overline{w}_{d,j}}^2 = \frac{1}{n_d - 1} \sum_{i \in J_d} (w_{ij} - \overline{w}_{d,j})^2.$$

Using these expressions for the mean and variance of $Y_{c,j}$, the Wald statistic is

$$\hat{z} = \frac{y_{c,j} - n_c \overline{w}_j}{\sqrt{p_c \sum_{d=1}^{C} n_d \big(S^2_{\overline{w}_{d,j}} + (1 - p_c) \overline{w}^2_{d,j} \big)}}, \tag{10}$$

and the null hypothesis is rejected if $P(Z > \hat{z}) < \alpha$.

Chapter 29: CIGSCR ITERATION

Together with the cluster association significance test, the iteration forms the semisupervised clustering framework in CIGSCR. The application of a hypothesis test determines which clusters should be used for classification, and an iteration works to produce a set of associated clusters with each class being represented by at least one associated cluster. This is accomplished by introducing new clusters that are likely to be associated, and when necessary, are associated with a class not already represented by a cluster.

In IGSCR, pure hard clusters are removed from the image that is clustered in subsequent iterations, focusing further refinement on clusters that failed to pass the purity test. K clusters are used for each iteration, presumably producing smaller clusters as less data is divided into the same number of clusters. The underlying assumption is that clusters that fail to pass the purity test could actually be composed of multiple clusters that would pass the purity test individually, and clustering the remaining data into K more clusters will reveal these smaller clusters. This method will not directly work on soft clusters as soft clusters cannot be removed simply by removing any sample associated with a pure cluster—all samples have a positive probability of belonging to any particular cluster.

In CIGSCR, unassociated clusters are targeted for refinement by using their information to create new clusters that will likely be associated. IGSCR is effectively locating smaller clusters that when combined to form a larger cluster would have been rejected. IGSCR accomplishes this by finding the same number of clusters (K) in the original dataset and then in successively smaller subsets of that original dataset. A similar approach that would locate smaller pure clusters in rejected clusters is "splitting" a cluster, employed by Ball and Hall [3] in ISODATA. Clusters are split by partitioning a cluster into two new clusters and recalculating new means. Soft clusters are represented by cluster means, and splitting a soft cluster would equate with replacing one cluster mean with two cluster means (calculated based on data associated with a cluster).

A cleaner algorithmic solution is to add one new cluster using information contained in the target cluster (the cluster that would be split), which effectively splits the cluster into two clusters. When using a clustering algorithm based on objective function (6), adding a new cluster guarantees a smaller function value (shown below) when $p = 2$. Using only the labeled samples belonging to the majority class (as determined in the cluster association significance test) to seed a new cluster would have the effect of pulling the new cluster toward those samples. Once another clustering iteration is completed, the targeted cluster would produce one cluster that is likely to be associated with the majority class and another cluster that retains relatively strong associations with all other classes. In CIGSCR, once the association significance test is performed, if at least one cluster is unassociated (and there are no unassociated classes), the cluster with the lowest value of $\hat{z}$ is used to generate a new cluster. The new cluster mean is determined using

$$U^{(K+1)} = \frac{\displaystyle\sum_{i \in \phi^{-1}(c_k)} w_{ik} X^{(i)}}{\displaystyle\sum_{i \in \phi^{-1}(c_k)} w_{ik}}, \tag{11}$$

where k is the cluster with the lowest value of $\hat{z}$, c_k is the majority class in cluster k, and recall that $\phi^{-1}(c)$ is the index set of labeled samples whose label is c. This formula also works when a class other than the majority class is used to seed a new cluster mean.

A shortcoming in IGSCR is that there is no guarantee that any clusters will be created and labeled with any particular class, and if a particular class is not represented by a cluster, the desired classification cannot be performed. In CIGSCR, this issue is addressed by adding a new cluster using information from a particular class if that class is not represented in the associated clusters. If a class c is not represented in the associated clusters, the cluster that is closest to being associated with c is used to generate a new cluster using (11) with $c_k = c$. The "closest" cluster is determined to be the cluster with the highest ratio of the average membership of class c to the average membership of the majority class.

When there are classes not represented by associated clusters and there are unassociated clusters, only one method can be used to determine the creation of a new cluster. If a cluster is unassociated, it is simply not used in classification. It is more important to have each class represented by the associated clusters than to refine an unassociated cluster, because the desired classification cannot be applied unless all classes are represented by associated clusters. Therefore adding a new cluster so that all classes will be represented takes precedence over adding a new cluster because an existing cluster is unassociated.

Finally, the theorem proving that adding one cluster mean will result in a smaller value of (6) is presented below.

Theorem: Given an integer $K > 0$, positive real numbers ρ_{ij}, $i = 1, \ldots, n$; $j = 1, \ldots, K+1$, defining a point $\rho \in \Re^{n \times K+1}$, and the objective function

$$J^{(K)}(\rho) = \sum_{i=1}^{n} \sum_{j=1}^{K} w_{ij}^2 \rho_{ij},$$

for K clusters where

$$w_{ij} = \frac{1/\rho_{ij}}{\displaystyle\sum_{k=1}^{K} 1/\rho_{ik}},$$

the objective function

$$J^{(K+1)}(\rho) = \sum_{i=1}^{n} \sum_{j=1}^{K+1} \hat{w}_{ij}^2 \rho_{ij},$$

for $K+1$ clusters where

$$\hat{w}_{ij} = \frac{1/\rho_{ij}}{\displaystyle\sum_{k=1}^{K+1} 1/\rho_{ik}},$$

satisfies

$$J^{(K+1)}(\rho) < J^{(K)}(\rho).$$

Proof: Note that the ρ_{ij} do not change with the addition of the $(K+1)$st cluster prototype, however $\hat{w}_{ij} < w_{ij}$ for $j < K+1$ because the denominator of $\hat{w}_{ij}$ has an additional term. Let $J_i^{(K)} = \sum_{j=1}^{K} w_{ij}^2 \rho_{ij}$ and $J_i^{(K+1)} = \sum_{j=1}^{K+1} \hat{w}_{ij}^2 \rho_{ij}$. It is sufficient to show that $J_i^{(K+1)} < J_i^{(K)}$ for each i to prove that $J^{(K+1)} < J^{(K)}$.

Let

$$S_1 = \sum_{k=1}^{K} 1/\rho_{ik} \qquad \text{and} \qquad S_2 = \sum_{k=1}^{K+1} 1/\rho_{ik}.$$

Then

$$w_{ij}^2 = \frac{(1/\rho_{ij})^2}{S_1^2} \qquad \text{and} \qquad \hat{w}_{ij}^2 = \frac{(1/\rho_{ij})^2}{S_2^2}.$$

$$\begin{aligned} J_i^{(K)} - J_i^{(K+1)} &= \sum_{j=1}^{K} \frac{(1/\rho_{ij})}{S_1^2} - \sum_{j=1}^{K+1} \frac{(1/\rho_{ij})}{S_2^2} \\ &= \frac{S_2^2 \sum_{j=1}^{K}(1/\rho_{ij}) - S_1^2 \sum_{j=1}^{K+1}(1/\rho_{ij})}{S_1^2 S_2^2}. \end{aligned}$$

Examining only the numerator in the previous term,

$$\begin{aligned} (S_1 + (1/\rho_{i,K+1}))^2 &\sum_{j=1}^{K}(1/\rho_{ij}) \\ &- S_1^2 \left(\sum_{j=1}^{K}(1/\rho_{ij}) + (1/\rho_{i,K+1}) \right) \\ &= (S_1 + (1/\rho_{i,K+1}))^2 S_1 - S_1^2(S_1 + (1/\rho_{i,K+1})) \\ &= S_1^3 + 2S_1^2(1/\rho_{i,K+1}) + S_1(1/\rho_{i,K+1})^2 \\ &\quad - S_1^3 - S_1^2(1/\rho_{i,K+1}) \\ &= S_1^2(1/\rho_{i,K+1}) + S_1(1/\rho_{i,K+1})^2 \\ &> 0 \end{aligned}$$

yielding

$$J_i^{(K+1)} < J_i^{(K)}.$$

Q.E.D.

Assuming that the clustering algorithm locates a local minimum point of the objective function, the combination of the clustering algorithm and this cluster prototype addition are guaranteed to move toward a smaller objective function value. If left unchecked, infinitely many clusters could be added, and the algorithm would continue to find smaller objective function values. The association significance test plays a crucial role in the termination of this iterative process. Once all clusters pass the association significance test and each class has at least one associated cluster, the iteration stops because the higher level objective has been met: clusters that significantly correspond to all classes have been located. The iteration also terminates when a maximum number of clusters is reached, and only those clusters that pass the association significance test are used for classification.

Chapter 30: DISTANCE FUNCTIONS

The previous theorem shows that for positive real numbers ρ_{ij}, $i = 1, \ldots, n$; $j = 1, \ldots, K+1$, and weights w_{ij} computed in a particular way, the addition of a cluster will result in a smaller value of the objective function $J(\rho) = \sum_{i=1}^{n} \sum_j w_{ij}^2 \rho_{ij}$. Although the soft clustering iteration for the objective function $J(\rho) = \sum_{i=1}^{n} \sum_j w_{ij}^2 \rho_{ij}$ is only guaranteed to converge when ρ_{ij} is Euclidean distance squared, [11] suggests that other functions may be used. The Euclidean distance squared is a special case of a radial function: $f : \Re^B \to \Re$ is *radial* if $f(x) = f(y)$ for $||x||_2 = ||y||_2$. Thus $\rho_{ij} = f(x^{(i)} - U^{(j)}) = ||x^{(i)} - U^{(j)}||_2^2$ is radial. Some alternative radial functions include

$$f(x) = \exp\bigl(||x||_2^q\bigr)$$

and

$$f(x) = ||x||_2^q$$

where $q \geq 1$ and $\rho_{ij} = f(x^{(i)} - U^{(j)})$. The advantage of using a radial function is that distances can be magnified so the difference between large and small cluster weights will be more extreme, approximating hard clustering.

None of the aforementioned metrics or radial functions influence the assignment of cluster weights based on the prelabeled points. Semisupervised clustering uses prior information to influence a clustering method. Although the association significance test and iteration are indirectly doing this, a modified objective function could directly use prior information to influence clusters. Consider the modified objective function component

$$J_i = \sum_{j=1}^{K} w_{ij}^2 \rho_{ij}(1 + \beta L_{ij}), \quad i = 1, \ldots, n,$$

where the term βL_{ij} is the penalty associated with assigning a labeled pixel to a cluster with a different associated label. Recall that $\phi(i) = c$ is the class label of the ith labeled pixel, and let $\phi(i) = \Omega \notin \{c_1, \ldots, c_C\}$ if the ith pixel is unlabeled,

$$C(j) = \begin{cases} c, & \text{if the } j\text{th cluster is associated with the} \\ & c\text{th class,} \\ \Omega, & \text{otherwise,} \end{cases}$$

$$L_{ij} = \begin{cases} 1, & \text{if } \phi(i) \neq \Omega,\ \phi(i) \neq C(j),\ C(j) \neq \Omega, \\ 0, & \text{otherwise.} \end{cases}$$

This objective function is still constrained subject to

$$\sum_{j=1}^{K} w_{ij} = 1.$$

Consider the problem of minimizing J_i with respect to W, for fixed i and all $U^{(j)}$ fixed. The Lagrangian of this constrained optimization problem is

$$\mathcal{L}(W, \lambda) = \sum_{j=1}^{K} \bigl[w_{ij}^2 \rho_{ij}(1 + \beta L_{ij})\bigr] - \lambda\Bigl(\sum_{j=1}^{K} w_{ij} - 1\Bigr)$$

giving the necessary optimality conditions

$$\frac{\partial \mathcal{L}}{\partial \lambda} = \sum_{j=1}^{K} w_{ij} - 1 = 0 \Rightarrow \sum_{j=1}^{K} w_{ij} = 1,$$

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial w_{ik}} &= 2w_{ik}\rho_{ik}(1+\beta L_{ik}) - \lambda = 0 \\ &\Rightarrow w_{ik} = \frac{\lambda}{2}\frac{1}{\rho_{ik}(1+\beta L_{ik})}.\end{aligned}$$

Summing w_{ik} over k gives

$$\begin{aligned}1 = \sum_{k=1}^{K} w_{ik} = \frac{\lambda}{2}\sum_{k=1}^{K} 1/\big(\rho_{ik}(1+\beta L_{ik})\big) &\Rightarrow \\ \frac{\lambda}{2} = \frac{1}{\displaystyle\sum_{k=1}^{K} 1/\big(\rho_{ik}(1+\beta L_{ik})\big)} &\Rightarrow \\ w_{ik} = \frac{1/\big(\rho_{ik}(1+\beta L_{ik})\big)}{\displaystyle\sum_{s=1}^{K} 1/\big(\rho_{is}(1+\beta L_{is})\big)}&.\end{aligned}$$

Therefore the distance function $f(x^{(i)} - U^{(j)}) = d_{ij} = \rho_{ij}(1+\beta L_{ij})$ can be substituted for ρ_{ij} in the CIGSCR algorithm to magnify the weights of pixels labeled with the cth class to clusters associated with the cth class.

Chapter 31: FORMAL CIGSCR ALGORITHM

Pseudocode for the complete CIGSCR algorithm including the association significance test and various distance functions is given below.

```
Algorithm CIGSCR
Input: $X$ (3-dimensional image)
$\phi^{-1}$ (set of $(row, col)$ indices for each class)
$Kinit$ (number of initial clusters)
$Kmax$ (maximum number of clusters)
$C$ (number of classes)
$\epsilon$ (convergence threshold)
$\alpha$ (Type-I error for one-sided hypothesis test)
$\beta$ (distance function penalty)
Output: $DR$ (decision rule classification)
$IS$ (iterative stacked classification)
  begin
    Initialize cluster means $U$ along the mean plus or
    minus the standard deviation of the image $X$;
    for $iteration := Kinit$ step 1 until $Kmax$ do
      begin
        $w := 0$; $convergence := 1$;
        while $convergence > \epsilon$ do
          begin
            $num := 0; denom := 0$;
            for $i := 1$ step 1 until $rows$ do
              for $j := 1$ step 1 until $cols$ do
                for $k := 1$ step 1 until $K$ do
                  begin
(*)                 $\hat{w}_{ij,k} := \frac{1/||X^{(ij)} - U^{(k)}||_2^2}{\sum_{l=1}^{K} 1/||X^{(ij)} - U^{(l)}||_2^2}$;
                    (update sums for mean calcs.)
                    $num^{(k)} := num^{(k)} + \hat{w}_{ij,k}^2 X^{(ij)}$;
                    $denom_k := denom_k + \hat{w}_{ij,k}^2$;
                  end
            (update cluster means)
            for $k := 1$ step 1 until $K$ do
              $U^{(k)} := \frac{num^{(k)}}{denom_k}$;
            $convergence := \max_{i,j,k} |w_{ij,k} - \hat{w}_{ij,k}|$;
            $w := \hat{w}$;
          end
        for $k := 1$ step 1 until $K$ do
          begin
            Determine majority class $c$ of cluster $k$;
            $c_k := c$;
  (**)      $Z_k := \frac{\sqrt{n_c}(\overline{w}_{c,k} - \overline{w}_k)}{s_{\overline{w}_k}}$;
          end
        if any class is not associated with a cluster then
          begin
            $c :=$ first unassociated class
            $k := \text{argmax}_k \frac{\overline{w}_{c,k}}{w_{c_k,k}}$
            $K := K + 1$
            $U^{(K)} = \frac{\sum_{ij \in \phi^{-1}(c)} w_{ij,k} X^{(ij)}}{\sum_{ij \in \phi^{-1}(c)} w_{ij,k}}$;
          end
        elseif (any$(Z_k < Z(\alpha), k = 1, \ldots, K)$ then
          begin
            $k := \text{argmin}_k Z_k$;
            $K := K + 1$;
            $U^{(K)} = \frac{\sum_{ij \in \phi^{-1}(c_k)} w_{ij,k} X^{(ij)}}{\sum_{ij \in \phi^{-1}(c_k)} w_{ij,k}}$;
          end
        else
          exit for loop;
        end
      end
    for $k := 1$ step 1 until $K$ do
      begin
        (initialize for covariance calcs.)
        $\Sigma_k := 0$;
        $denom_k := 0$;
      end
    (IS classification)
    for $i := 1$ step 1 until $rows$ do
      for $j := 1$ step 1 until $cols$ do
        begin
          $csum := 0$;
          for $k := 1$ step 1 until $K$ do
            if $(Z_k > Z(\alpha))$ then
              $csum_{c_k} := csum_{c_k} + w_{ij,k}$;
          for $c := 1$ step 1 until $C$ do
            $IS_{ij,c} := \frac{csum_c}{\sum_{k=1}^{C} csum_k}$;
```

```
            (calculate covariance matrices)
            for $k := 1$ step 1 until $K$ do
              begin
                $\Sigma_k := \Sigma_k + w_{ij,k}$
                 $\cdot(X^{(ij)} - U^{(k)})(X^{(ij)} - U^{(k)})^T$;
                $denom_k := denom_k + w_{ij,k}$;
              end
          end
      for $k := 1$ step 1 until $K$ do
        $\Sigma_k := 1/denom_k \cdot \Sigma_k$;
      (DR classification)
      for $i := 1$ step 1 until $rows$ do
        for $j := 1$ step 1 until $cols$ do
          begin
            $csum := 0$;
            for $k := 1$ step 1 until $K$ do
              if $(Z_k > Z(\alpha))$ then
                begin
                  $p := \frac{2e^{-\frac{1}{2}(X^{(ij)}-U^{(k)})^T\Sigma_k^{-1}(X^{(ij)}-U^{(k)})}}{\pi^{B/2}|\Sigma_k|^{\frac{1}{2}}}$;
                  $csum_{c_k} := csum_{c_k} + p$;
                else
                  $csum_{c_k} := 0$;
                end
              for $c := 1$ step 1 until $C$ do
                $DR_{ij,c} := \frac{csum_c}{\sum_{k=1}^{C} csum_k}$;
          end
    end
```

Note that in place of the distance function used in line (*) of the above algorithm, one of the radial functions of Euclidean distance or a distance function with a penalty (β) could be used. Also, (10) could be used in place of the less sophisticated Z_k calculation in line (**).

Chapter 32: CIGSCR EXPERIMENTAL RESULTS AND DISCUSSION

The first dataset used to obtain experimental results for IGSCR and CIGSCR is a mosaicked Landsat Enhanced Thematic Mapper Plus (ETM+) satellite image taken from Landsat Worldwide Reference System (WRS) path 17, row 34, located in Virginia, USA, shown in Figure 47. This image, hereafter referred to as VA1734, was acquired on November 2, 2003 and consists largely of forested, mountainous regions, and a few developed regions that are predominantly light blue and light pink in Figure 47. Figure 47 contains a three color representation of VA1734 where the red color band in Figure 47 corresponds to the near infrared wavelength in VA1734, the green color band in Figure 47 corresponds to the red wavelength in VA1734, and the blue color band in Figure 47 corresponds to the green wavelength in VA1734. Figure 48 contains a zoomed area of interest.

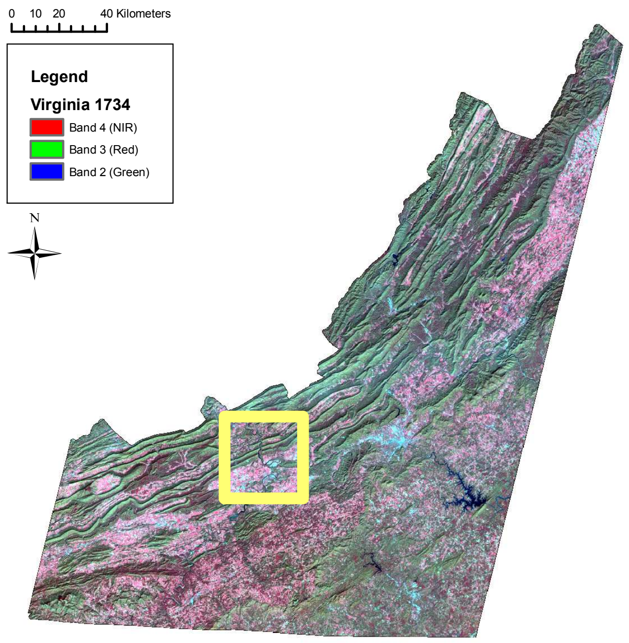


Figure 47. Landsat ETM+ path 17/row 34 over Virginia, USA with area of interest highlighted.

The training data for this image was created by the interpretation of point locations from a systematic, hexagonal grid over Virginia Base Mapping Program (VBMP) true color digital orthophotographs. A two class classification was performed (forest/nonforest), and classification parameters and results are given in Table 9 (DR classification) and Table 10 (IS/IS+ classification). Classification images for this dataset are given in Figures 49 though 54.

Validation data in the form of point locations at the center of USDA Forest Service Forest Inventory and Analysis (FIA) ground plots were used to assess the accuracy of this classification. Since these validation data are typically used to evaluate crisp classifications, only homogeneous FIA plots were used (either 100 percent forest or nonforest), and these plots were obtained between 1997 and 2001. Accuracy was assessed based on an error matrix where classification results for specific points (not included in the training data set) are

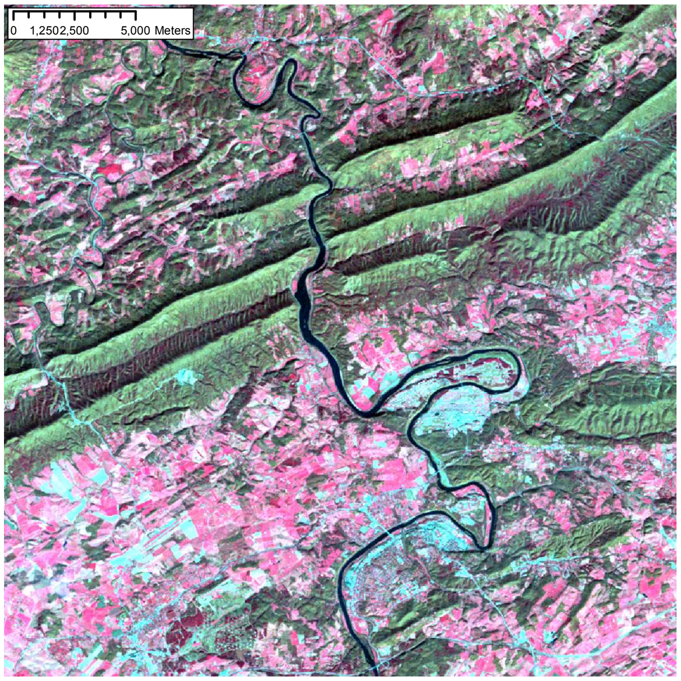


Figure 48. Landsat ETM+ path 17/row 34 over Virginia, USA area of interest.

compared against known class values. The accuracies reported in Tables 9–12 were obtained by first converting all soft classifications to hard classifications for the purpose of comparing hard classification values to hard ground truth values. The classification results reported in Tables 9–12 used 10, 15, 20, and 25 initial clusters for IGSCR and CIGSCR. Experimental runs of IGSCR used homogeneity thresholds (test probabilities of observing the majority class in a particular cluster) of .5 and .9, with $\alpha = .01$ for all IGSCR classifications. A threshold of .9 would indicate a homogeneous cluster, but a threshold of .5 is perhaps more analogous to the new association significance test used in CIGSCR. Experimental runs of CIGSCR used traditional Euclidean distance squared in addition to two proposed radial functions $f(X^{(i,j)} - U^{(k)}) = ||X^{(i,j)} - U^{(k)}||_2^4$ and $f(X^{(i,j)} - U^{(k)}) = \exp(||X^{(i,j)} - U^{(k)}||_2)$. For all reported CIGSCR runs, $\alpha = .0001$ (values of $\hat{z}$ tend to be high for the association significance test). All reported CIGSCR classifications used hypothesis test (10). Only three out of 24 total CIGSCR classifications reported in this thesis were different using (9) and (10), and the difference in resulting classification accuracies was not significant and did not show that one test consistently resulted in higher classification accuracies than the other test. Values of $\hat{z}$ are slightly smaller using (10) than (9), resulting in more potential for cluster refinement. Additionally, the distance function with penalty was used in classification, although results are not reported in Tables 9 and 10 because incorporating the penalty into the distance function did not increase classification accuracies in any experimental runs. Large values of β produced less accurate classification results. Finally, classification was performed using just clustering without the semisupervised framework to evaluate the effect of the combination of the association significance test and iteration in CIGSCR on classification accuracies.

The second dataset used to obtain experimental results for IGSCR and CIGSCR is a hyperspectral image of the Appomattox Buckingham State Forest in Virginia, USA. The

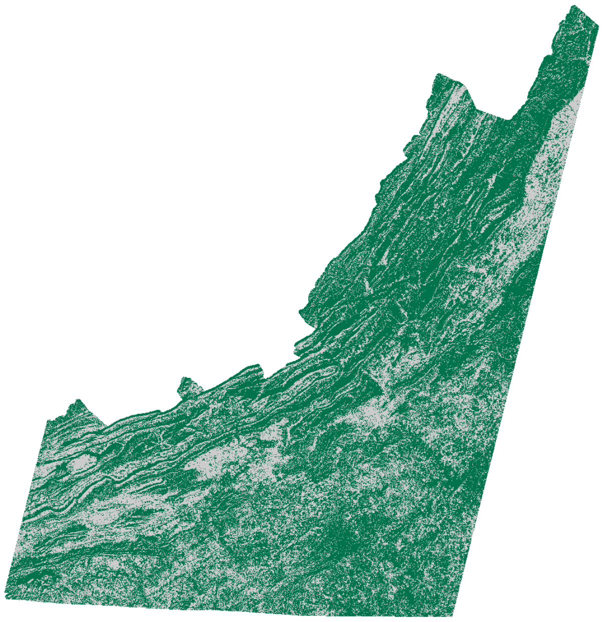


Figure 49. IGSCR DR classification of VA1734(Landsat ETM+ path 17/row 34) using 10 initial clusters and a homogeneity threshold of 90%.

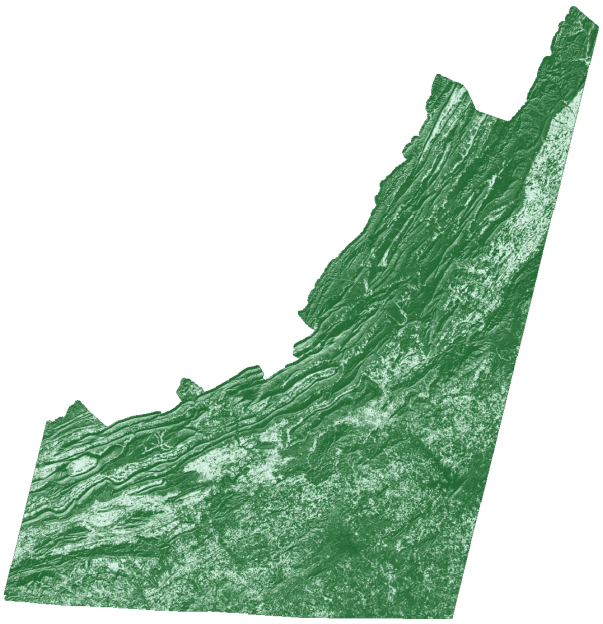


Figure 50. CIGSCR DR classification of VA1734 (Landsat ETM+ path 17/row 34) using 10 initial clusters and Euclidean distance squared.

AVIRIS 224-band, low-altitude flight lines were acquired in the winter of 1999 and ranged from approximately 400-2500nm (10nm spectral resolution) with 3.4m spatial resolution [91]. The AVIRIS data were geometrically and radiometrically corrected (to level 1B at-

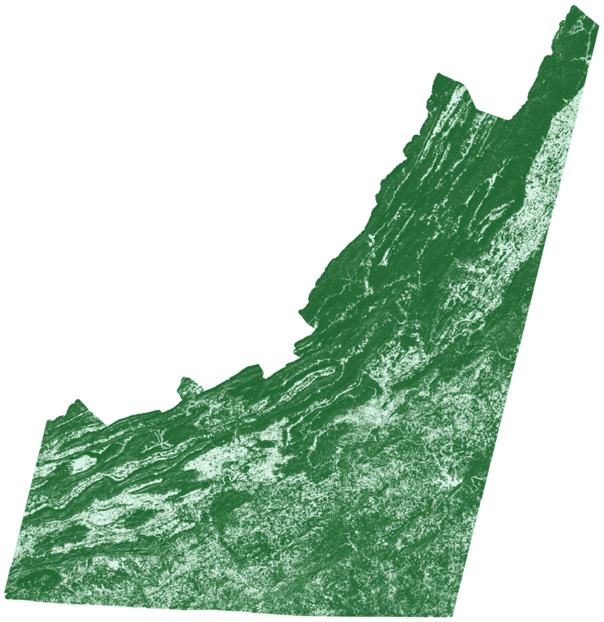


Figure 51. CIGSCR DR classification of VA1734 (Landsat ETM+ path 17/row 34) using 10 initial clusters and Euclidean distance to the fourth power.

sensor radiance, units of microwatts per square centimeter per nanometer per steradian) by the Jet Propulsion Laboratory (JPL; Pasadena, California, USA). The three flight lines used for this study were registered (8–12 control points per flight line) to an existing 0.5m orthophoto of the area. Resampling resulted in root mean square errors (RMSE) ranging between 0.23 and 0.24 pixels [91].

Training data were acquired by collecting 142 field locations [91]surrounded by homogeneous areas of single pine species (64 loblolly (*Pinus taeda*), 30 shortleaf (*Pinus echinata*), and 48 Virginia pine (*Pinus virginiana*)) with differentially corrected global positioning system (GPS) coordinates. These locations were used in a region growing algorithm to obtain a sufficient number of points for training and validation, and nonpine training data were acquired using knowledge of the area and maps of known stands in the region. The image (shown in Figure 55 and hereafter referred to as ABSF) contains various tree stands that include the three species of pines listed above, hardwoods, and mixed (evergreens and hardwoods).

400 points were randomly selected to serve as validation data for these four classes (loblolly, shortleaf, and Virginia pines, and nonpine). Classification results for these data are reported in Tables 11 and 12, Figure 56 contains the IGSCR IS classification image using 25 initial clusters and a homogeneity threshold of .5, and Figures 57a–d contain the CIGSCR IS classification images using 10 initial clusters and Euclidean distance to the fourth power. Classifications were run using the same parameters as classifications reported in Tables 9 and 10. An asterisk (*) indicates that the classification failed because at least one class had no associated clusters. Tables 13 and 14 report the number of pure clusters (IGSCR), and the number of clusters produced and number of associated clusters (CIGSCR).

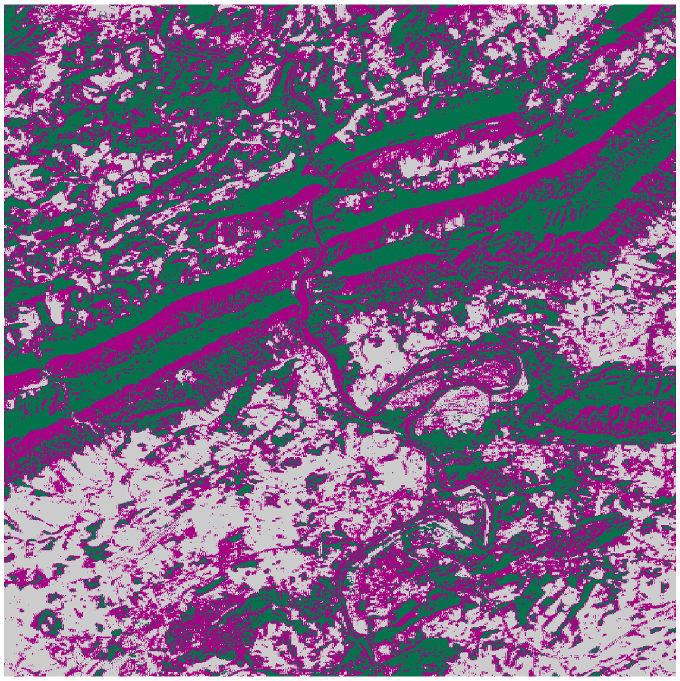


Figure 52. IGSCR IS classification of VA1734 (Landsat ETM+ path 17/row 34) using 10 initial clusters and a homogeneity threshold of 90%.

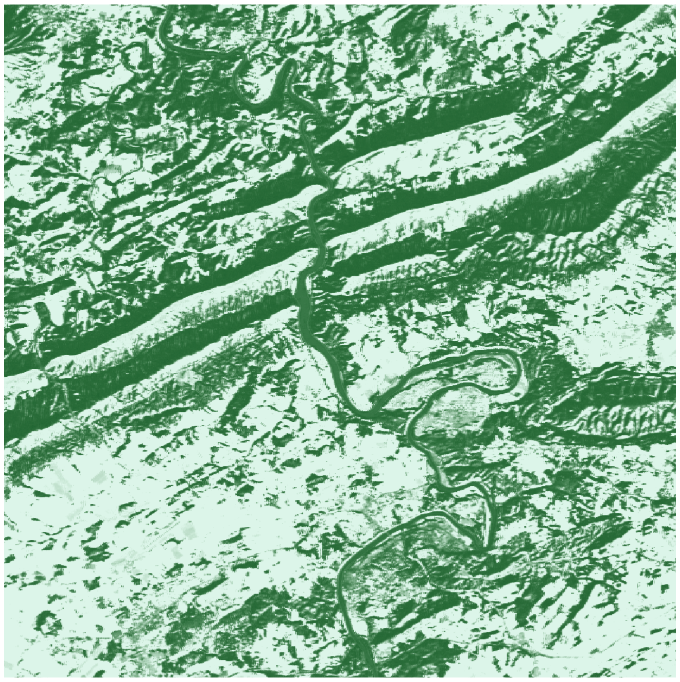


Figure 53. CIGSCR IS classification of VA1734 (Landsat ETM+ path 17/row 34) using 10 initial clusters and Euclidean distance squared.

32.1 Discussion.

The soft clustering and soft classification in CIGSCR can result in qualitatively different

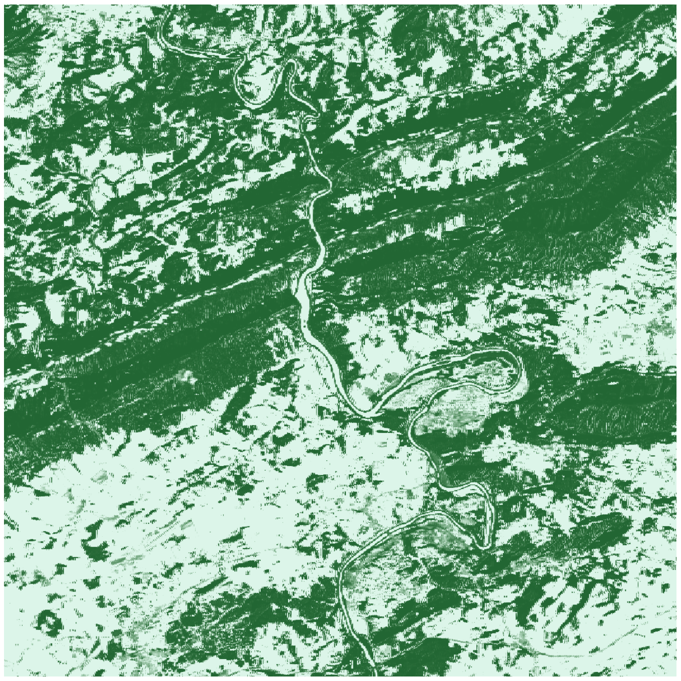


Figure 54. CIGSCR IS classification of VA1734 (Landsat ETM+ path 17/row 34) using 10 initial clusters and Euclidean distance to the fourth power.

Table 9. IGSCR and CIGSCR decision rule (DR) classification accuracies for VA1734..

no. init.	IGSCR ($\alpha = .01$)		CIGSCR ($\alpha = .0001$)			clustering
clusters	$p = .5$	$p = .9$	$\rho = \|\| \cdot \|\|_2^2$	$\rho = \|\| \cdot \|\|_2^4$	$\rho = e^{\|\| \cdot \|\|_2}$	(no iteration)
10	85.81	75.49	88.74	87.07	87.70	72.26
15	88.22	74.56	80.50	88.53	86.97	73.72
20	84.78	89.57	79.87	89.68	88.74	76.54
25	87.49	84.25	81.44	89.47	88.74	77.58

Table 10. IGSCR iterative stacked plus (IS+) and CIGSCR iterative stacked (IS) classification accuracies for VA1734..

no. init.	IGSCR ($\alpha = .01$)		CIGSCR ($\alpha = .0001$)			clustering
clusters	$p = .5$	$p = .9$	$\rho = \|\| \cdot \|\|_2^2$	$\rho = \|\| \cdot \|\|_2^4$	$\rho = e^{\|\| \cdot \|\|_2}$	(no iteration)
10	68.30	75.39	83.63	84.67	85.09	72.26
15	86.34	74.56	76.96	86.03	85.19	72.99
20	84.46	88.95	75.60	85.40	86.86	76.85
25	66.63	83.94	78.52	88.32	87.28	76.75

classifications than IGSCR. Even when the final classifications are similar, CIGSCR provides more information through soft classification. The soft classifications in Figures 50 and 51 compared to the hard classification in Figure 49 show that even when the hard clustering/classification in IGSCR and the soft clustering/classification in CIGSCR identify the same general regions as likely to be forest or likely to be nonforest, the soft classifications

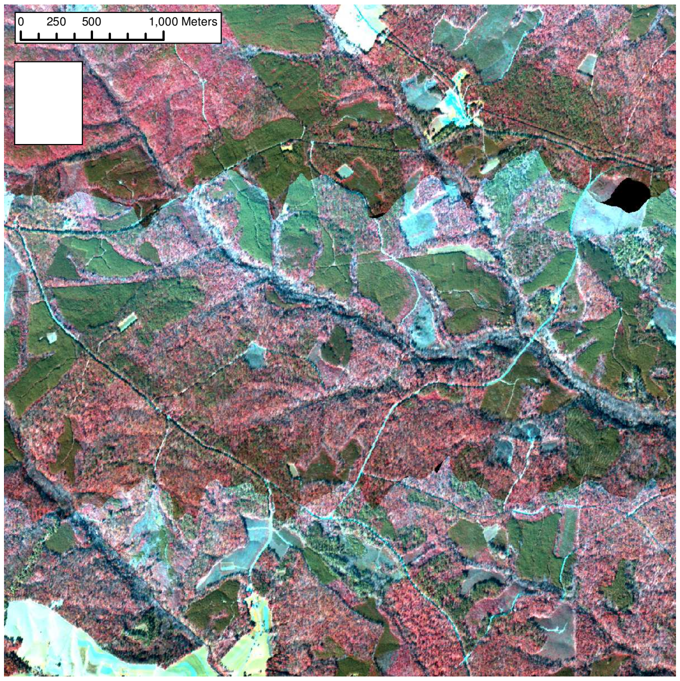


Figure 55. AVIRIS image (three flight lines) taken over Appomattox Buckingham State Forest in Virginia, USA.

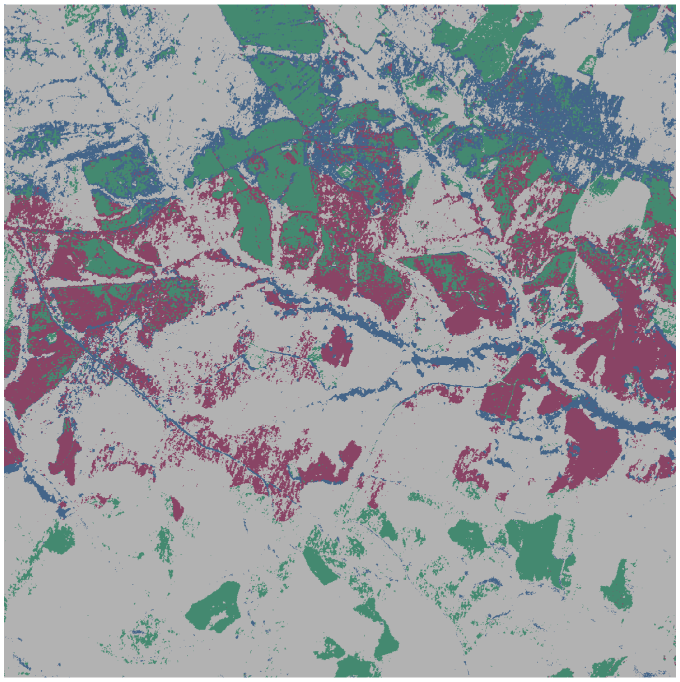


Figure 56. IGSCR IS classification of ABSF (3 flight lines of AVIRIS imagery).

in Figures 50 and 51 provide extra information relating to how strongly a particular sample is forest or nonforest. The dark green and dark brown colors indicate a high probability of forest and nonforest, respectively. Lighter shades of both colors indicate lower probabilities

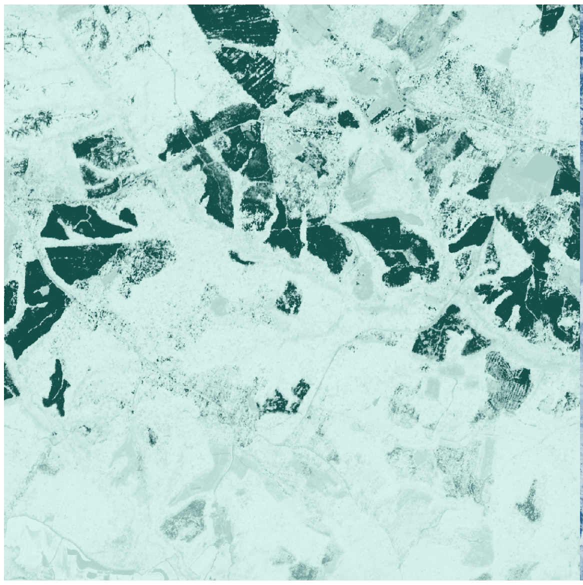


Figure 57a. CIGSCR IS classification (loblolly pines) of ABSF (3 flight lines of AVIRIS imagery).

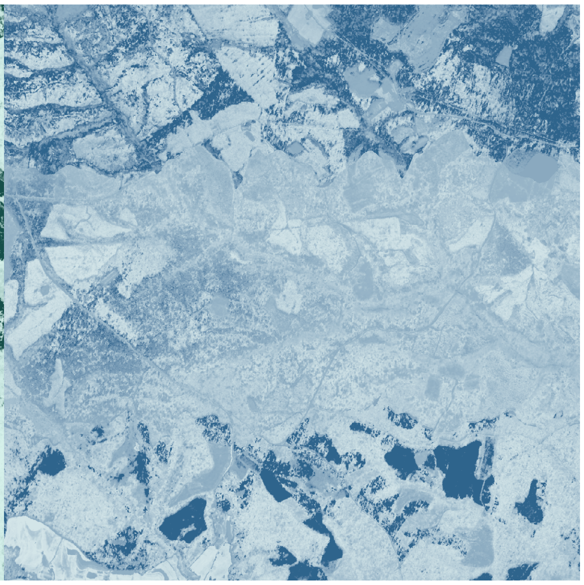


Figure 57b. CIGSCR IS classification (shortleaf pines) of ABSF (3 flight lines of AVIRIS imagery).

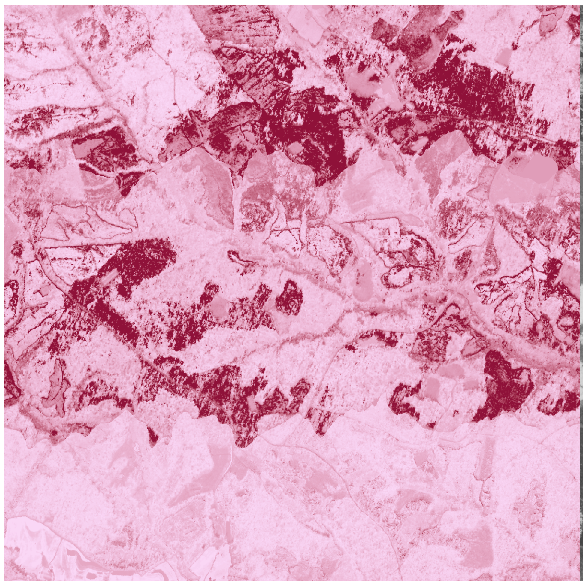


Figure 57c. CIGSCR IS classification (Virginia pines) of ABSF (3 flight lines of AVIRIS imagery).

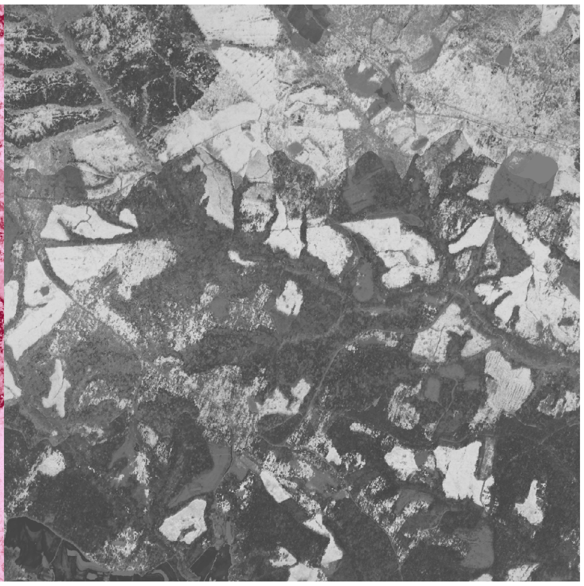


Figure 57d. CIGSCR IS classification (nonpine) of ABSF (3 flight lines of AVIRIS imagery).

of membership in respective classes, and the beige regions indicate that the probabilities of that region being forest or nonforest are almost equal. The classifications in Figures 56 and 57a–d show that in addition to providing more information, CIGSCR can produce qualitatively different classifications than IGSCR. The classifications present in Figures 56 and 57a–d are the IS classifications that result from clustering, showing that soft clustering in CIGSCR produces different clustering and classification than the hard clustering in IGSCR.

Table 11. IGSCR and CIGSCR decision rule (DR) classification accuracies for ABSF..

no. init.	IGSCR ($\alpha = .01$)		CIGSCR ($\alpha = .0001$)			clustering
clusters	$p = .5$	$p = .9$	$\rho = \|\|\cdot\|\|_2^2$	$\rho = \|\|\cdot\|\|_2^4$	$\rho = e^{\|\|\cdot\|\|_2}$	(no iteration)
10	83.50	*	47.50	79.50	72.50	*
15	*	*	62.50	83.50	79.75	*
20	*	*	66.75	73.50	74.25	*
25	51.00	51.00	63.00	75.00	78.75	*

Table 12. IGSCR iterative stacked plus (IS+) and CIGSCR iterative stacked (IS) classification accuracies for ABSF..

no. init.	IGSCR ($\alpha = .01$)		CIGSCR ($\alpha = .0001$)			clustering
clusters	$p = .5$	$p = .9$	$\rho = \|\|\cdot\|\|_2^2$	$\rho = \|\|\cdot\|\|_2^4$	$\rho = e^{\|\|\cdot\|\|_2}$	(no iteration)
10	83.75	*	51.75	84.50	72.75	*
15	*	*	51.00	84.50	83.25	*
20	*	*	51.00	84.00	81.50	*
25	91.00	75.25	51.00	76.75	83.00	*

Table 13. For VA17 IGSCR, number of pure clusters. For VA17 CIGSCR, the pairs (a,b) = (number of clusters produced, number of associated clusters)..

no. init.	IGSCR		CIGSCR		
clusters	$p = .5$	$p = .9$	$\rho = \|\|\cdot\|\|_2^2$	$\rho = \|\|\cdot\|\|_2^4$	$\rho = e^{\|\|\cdot\|\|_2}$
10	19	6	15,13	11,11	12,12
15	15	6	20,16	20,19	20,20
20	20	18	25,21	21,21	24,24
25	52	17	30,25	30,28	30,29

Table 14. For ABSF IGSCR, number of pure clusters. For ABSF CIGSCR, the pairs (a,b) = (number of clusters produced, number of associated clusters)..

no. init.	IGSCR		CIGSCR		
clusters	$p = .5$	$p = .9$	$\rho = \|\|\cdot\|\|_2^2$	$\rho = \|\|\cdot\|\|_2^4$	$\rho = e^{\|\|\cdot\|\|_2}$
10	16	8	15,15	10,10	11,11
15	14	11	20,19	15,15	15,15
20	19	9	25,24	20,20	20,20
25	23	15	30,29	25,25	26,26

The regions identified by CIGSCR as being likely to contain individual pine species are different from the regions identified by IGSCR, although both algorithms identified similar nonpine regions.

Based on accuracies reported in Tables 9 and 10, CIGSCR is less sensitive to the number of initial clusters than IGSCR, especially when the alternative radial functions are used. As shown in Tables 9 and 10, IGSCR can be sensitive to the number of initial clusters and the homogeneity threshold. The set of clusters ultimately used for classification in IGSCR is directly affected by the number of initial clusters and the homogeneity test, and furthermore, when all clusters fail the homogeneity test, the iteration terminates and no more clusters are found. The number of clusters used for classification can vary widely depending on the number of iterations completed as each iteration potentially produces several pure clusters. The low accuracies reported for the IGSCR IS+ classifications in Table 10 occur when a small number of iterations occurs, which can be greatly influenced by the number of initial clusters and the homogeneity test. The classification accuracies reported for CIGSCR in Tables 9 and 10 are more consistent as CIGSCR does not have the same sensitivity issues. First, the association significance test no longer requires a user input threshold like the homogeneity test. The homogeneity test evaluates the observed values against a user supplied probability of observing a specific class (within a cluster), but the association significance test determines if the average cluster memberships per class are statistically significantly different (requiring no user specified probability). Secondly, the iteration in CIGSCR is fundamentally different from the iteration in IGSCR. While each iteration in IGSCR locates multiple clusters, each iteration in CIGSCR adds one additional cluster, and terminating this iteration potentially excludes many fewer clusters from the final classification than terminating the iteration in IGSCR (especially when few iterations occur). As classification methods are already sensitive to training data and clustering methods are sensitive to initial prototype locations, classifications being sensitive to fewer parameters is a desirable property.

The CIGSCR classifications shown in Figures 50, 51, 53, and 54 experimentally validate the discussion in Chapter 30 that radial functions magnify the difference between the largest and smallest cluster weights and will more closely approximate hard clustering. The classifications based on clustering with Euclidean distance to the fourth power have significantly fewer samples with almost equal probabilities of being in either class (corresponding to the medium green color in the classification images). The classifications based on clustering with an exponential function of Euclidean distance (not pictured) are even closer to hard classification. Some medium green areas remain in Figures 51 and 54, indicating that although classifications based on these functions become more like hard classifications, in practice these classifications retain desirable properties of soft classification. Based on accuracies reported in Tables 9 and 10, these CIGSCR classifications with alternative radial functions are often the most accurate classifications for a given number of initial clusters. CIGSCR with alternative radial functions is accurate, can approximate hard classification when hard classification is desired, still provides more information than strict hard classification, and is less sensitive to input parameters than IGSCR.

All classification methods can be expected to perform poorly when training data are insufficient (samples within the dataset are not represented in the training set). This is especially true in IGSCR where spectrally pure hard clusters containing multiple training samples must be located in order for samples to be labeled with that particular class. In the VA1734 dataset, an example of a spectral class with insufficient training data is water, and although water is technically nonforest, water is often classified as forest because water and forest are spectrally similar in certain wavelength regions. This is the case in Figure 53 where the New River running vertically through the zoomed area of interest has been identified as forest. In Figure 52, this region is “unclassified” meaning that these pixels are not part of a pure cluster as expected (few or no water training samples are identified for this image/training dataset). Another misclassification occurs as a result of shadows in the forested mountains running diagonally in the upper half on the zoomed image (Figures 52, 53, and 54). Figure 52 indicates a likelihood that there is insufficient training data for these regions. Ultimately these water and shadow regions are misclassified using the decision rule in IGSCR (not pictured), and these regions are classified incorrectly using CIGSCR with Euclidean distance squared. However, notice in Figure 54 that the CIGSCR IS using Euclidean distance to the fourth power correctly classified the river and the shadow regions. With soft clustering, different clusters were formed, allowing these features to potentially be correctly placed in similar clusters, even though these clusters likely contained small percentages of the training data. In this case, it is potentially useful to know that these features are unclassified (in IGSCR) allowing for modification of the training data, and unfortunately CIGSCR does not have this capability. However, when more training samples are not available, CIGSCR can potentially provide a better estimate of the correct class for these data that are not well represented in the training data (although this is obviously not guaranteed as CIGSCR using two different distance functions produced different classification results). Also of interest is that the uncertainty in the soft classifications does not necessarily match the unclassified regions in Figure 52. There does not appear to be a correlation between samples that are not part of pure clusters in IGSCR and samples that may belong to multiple classes in CIGSCR.

The accuracies reported for the classification of ABSF tend to be lower than the classification accuracies reported for VA1734, which is reasonable considering the classification of ABSF is attempting to discriminate between spectrally similar pine species, ABSF is noisy, and ABSF contains several heterogeneous areas, making training difficult. Also note that the VA1734 DR classifications were almost always more accurate than corresponding IS classifications, but ABSF DR classifications are often less accurate than corresponding IS classifications. All ABSF classifications (IGSCR DR and IS and CIGSCR DR and IS) reasonably separated pines from nonpines, but IGSCR and CIGSCR differed in the identification of individual pines species. Both classification methods identified individual pines in mixed hardwood/pine stands in the top left corner of the image (Figures 56 and 57a–d). A visual inspection of the classification images reveals that IGSCR and CIGSCR classifications disagree on loblolly (IGSCR has underestimated those stands) and shortleaf (both overestimated). IGSCR incorrectly picked out patches of shortleaf along the "veins" of the image, and both classifications overestimated Virginia pines.

Another potential advantage of CIGSCR with an alternative radial function is the ability to locate clusters associated with classes, even when there is overlap between classes or there is a small amount of training data for a class. IGSCR failed to locate enough pure clusters to perform classification, indicated by an asterisk in Tables 11 and 12, in most ABSF classification attempts. CIGSCR using Euclidean distance squared produced classifications, although the accuracies are low. CIGSCR using alternative radial functions performed reasonable classifications no matter the number of initial clusters. In highly heterogeneous sites like this where limited training data is available for multiple classes, IGSCR has difficulty locating pure clusters. Since multiple classes are spectrally similar, soft clustering allows for small differences between classes in a cluster to be detected. Hard clusters containing one species would be likely to contain a significant amount of the other species, and would therefore fail the hypothesis test (for reasonable p and α. With soft clustering, portions of both species would be attributed to a soft cluster, but if there is statistical significance of the difference in the memberships of the species, the cluster can be associated and used for training purposes. Furthermore, soft clustering allows for alternative functions to be used to determine cluster assignments. Recall that these radial functions magnify the difference between small and large probabilities, allowing clusters containing these less well represented classes to be formed and allowing samples to have high probabilities of belonging to those clusters.

Finally, perhaps the most important question about this semisupervised clustering scheme is whether using the combination of the association significance test and the iteration improves the clustering for the purposes of classification. Each cluster is labeled with the class that has the highest average membership in the cluster. Observe in experimental runs in Tables 9 and 10 that **all** classification accuracies using just clustering are lower than corresponding classification accuracies using CIGSCR with Euclidean distance. In Tables 11 and 12, iterative refinement was necessary to locate enough clusters (such that each class was represented by at least one cluster) for classification using Euclidean distance squared. Accuracies are much higher using alternative distance functions, but little or no iterative refinement was used. Based on the available results in Tables 9–12, the semisupervised clustering scheme in CIGSCR improves classification accuracies when training data are available to influence clustering.

Chapter 33: CIGSCR CONCLUSIONS

This work introduced a hypothesis test that can be used to evaluate the suitability of soft clusters for classification. This thesis presented a continuous analog to IGSCR that rejects and refines clusters to automatically classify a remotely sensed image based on informational class training data. This new algorithm addressed specific challenges presented by remotely sensed data including large datasets (millions of samples), relatively small training datasets, and difficulty in identifying spectral classes. The resulting classifications are fundamentally different from IGSCR (the discrete predecessor to CIGSCR) classifications, even when converting the CIGSCR soft classifications to hard classifications. CIGSCR has many advantages over IGSCR, such as the ability to produce soft classification, less sensitivity to certain input parameters, ability to use alternative distance functions that often produce more accurate classifications, potential to correctly classify regions that are not amply represented in training data, and a better ability to locate clusters associated with all classes. The semisupervised clustering framework within CIGSCR has been shown here to improve classification accuracies over clustering alone. This semisupervised clustering framework could be incorporated into many classification algorithms that use clustering.

The proposed hypothesis tests based on (9) and (10) showed that differences in soft cluster memberships are often statistically significant. Similarly, the standard Euclidean distance and radial functions of Euclidean distance produced better clusters (evaluated by final classification accuracy) than a proposed distance function explicitly imposing penalties for the assignment of labeled data to labeled clusters when a label mismatch would have occurred. The radial functions used in CIGSCR resulted in consistently accurate classifications.

The highly automated CIGSCR classification algorithm is a contribution to the remote sensing community that has few if any automated semisupervised soft classification algorithms analogous to the many automated semisupervised hard classification algorithms that exist. Future work includes using this soft classifier for many applications of classification in remote sensing.

Chapter 34: CONCLUSIONS AND FUTURE WORK

This thesis focused on the study of automated, hybrid classification algorithms that produced hard and soft classifications and associated data reduction and noise removal algorithms. This work introduced a parallel version of the well-known IGSCR classification algorithm that significantly reduced the execution time necessary for IGSCR classifications, enabling further study of IGSCR. In a further effort to reduce the execution time necessary for classification, this work introduced a new data reduction technique based on the SVD. Experiments showed that SVD-based data reduction ultimately led to reduced classification times without significant reductions in classification accuracy. SVD-based data reduction performed better than another commonly used technique based on principal components analysis. A third contribution of this thesis was a data reduction technique coupled with noise removal in an adaptive filter based on the MNF. The removal of noise using the two adaptive filters developed for this thesis resulted in significant classification accuracy increases without drastic signal degradation that would occur using large uniform filters. Finally, a soft classification method based on IGSCR, CIGSCR, was introduced. CIGSCR is capable of producing accurate classifications in certain situations where IGSCR does not.

Future work includes the development of a parallel version of CIGSCR. CIGSCR was designed with the ultimate goal of a parallel version, and underlying algorithms and methods were selected only if they are scalable (such as fuzzy k-means). Using a parallel version of CIGSCR will make further study of the algorithm possible. New applications that are well-suited for soft classification should be attempted using CIGSCR, and parameter studies will be necessary for different types of applications and images. Finally, developing CIGSCR highlighted some fundamental issues in IGSCR, including the iteration. Some new methods used in CIGSCR should be adapted to IGSCR, hopefully resulting in improvements to the algorithm.

References

[1] G. M. Amdahl, “Validity of the single processor approach to achieving large scale computing capabilities,” In *AFIPS Conference Proceedings*, Vol. 30, pp 483-485, 1967.

[2] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen, *LAPACK Users' Guide*, SIAM, Philadelphia, 1992, 235pp.

[3] G. H. Ball and D. J. Hall, “A novel method of data analysis and pattern classification,” Techincal Report AD 699616 Stanford Research Institute, Menlo Park, CA, 9pp, 1965.

[4] S. Basu, A. Banerjee, and R.J. Mooney, “Active semi-supervision for pairwise constrained clustering,” in *Proc. of the SIAM International Conference on Data Mining*, SIAM, Philadelphia, 2004.

[5] M. E. Bauer, T. E. Burk, A. R. Ek, P. R. Coppin, S. D. Lime, T. A. Walsh, D. K. Walters, W. Befort, and D. F. Heinzen, “Satellite inventory of Minnesota forest resources,” *Photogrammetric Engineering & Remote Sensing*, 60(3), 287-298, 1994.

[6] W. A. Bechtold and C. T. Scott, “The Forest Inventory and Analysis Plot Design,” In The Enhanced Forest Inventory and Analysis Program – National Sampling Design and Estimation Procedures, W. A. Bechtold and P. L. Patterson, editors. USDA Forest Service Southern Research Station, General Technical Report SRS-80, pp. 27-42, 2005.

[7] M. Belkin and P. Niyogi, “Semisupervised learning on Riemannian manifolds,” *Machine Learning – Special Issue on Cluster*, 56 (2004), 209-239.

[8] A.M. Bensaid, L.O. Hall, J.C. Bezdek, and L.P. Clarke, “Partially supervised clustering for image segmentation,” *Pattern Recognition*, 29(1996), 859-871.

[9] A. J. Bernstein, “Analysis of programs for parallel processing,” *IEEE Transactions on Computers*, 746-757, Oct. 1966.

[10] J. Bezdek, “Fuzzy mathematics in pattern classification,” PhD Thesis, Cornell University, Ithaca, NY, 1974.

[11] J.C. Bezdek, “A convergence theorem for the fuzzy ISODATA clustering algorithms,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-2(1), 1–8.

[12] J. Bigün, “Unsupervised feature reduction in image segmentation by local transforms,” *Pattern Recognition Letters*, 14, 573-583, 1993.

[13] M. Bilenko, S. Basu, and R.J. Mooney, “Integrating constraints and metric learning in semisupervised clustering,” in *Proc. of the 21st International Conference on Machine Learning*, Palo Alto, CA: Morgan Kaufmann, 2004.

[14] J.W. Boardman and F. A Kruse, “Automated spectral analysis: a geological example using AVIRIS data, North Grapevine Mountains, Nevada,” in *10th Thematic Conference on Geologic Remote Sensing*, Ann Arbor: Environmental Research Institute of Michigan, Vol 1: pp. 407–418.

[15] A. Bouchachia and W. Pedrycz, “Data clustering with partial supervision,” *Data Mining and Knowledge Discovery*, 12, 47–78, 2006.

[16] L. Bruzzone and D. F. Prieto, “Unsupervised Retraining of a Maximum Likelihood Classifier for the Analysis of Multitemporal Remote Sensing Images,” *IEEE Trans. on Geosciences and Remote Sensing*, 39(2), 456-460, 2001.

[17] L. Bruzzone, M. Chi, and M. Marconcini, “A novel transductive SVMs for semi- supervised classification of remote-sensing images,” *IEEE Transactions on Geoscience and Remote Sensing*, 44(2006), 3363-3373.

[18] J. Byeungwoo and D. Landgrebe, “Partially Supervised Classification Using Weighted Unsupervised Clustering,” *IEEE Trans. on Geosciences and Remote Sensing*, 37(2), 1073-1079, 1999.

[19] J. B. Campbell, *Introduction to Remote Sensing*, The Guilford Press, 2002, 621pp.

[20] G. Camps-Valls, T.V. Bandos Marsheva, and D. Zhou, “Semisupervised graph- based hyperspectral image classification,” *IEEE Transactions on Geoscience and Remote Sensing*, 45(2007), 3044-3054.

[21] A.B. Carlson, P.B. Crilly, and J.C. Rutledge, *Communication Systems: An Introduction to Signals and Noise in Electrical Communication, fourth ed.*, New York: McGraw-Hill, 850 pp., 2002.

[22] G. Casella and R.L. Berger, *Statistical Inference (second ed.)*, Duxbury, Pacific Grove, CA, 2002, 660pp.

[23] W. C. Chang, “On using principal components before separating a mixture of two multivariate normal distributions,” *Applied Statistics*, 32, 267-275, 1983.

[24] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-Supervised Learning*, MIT Press, Cambridge, MA, 2006.

[25] C. Clark, A. F. Clark, “Spectral identification by singular value decomposition,” *International Journal of Remote Sensing*, 19(12), 2317-2329, 1998.

[26] R.R. Colditz, T. Wehrmann, M. Bachmann, K. Steinnocher, M. Schmidt, G. Strunz, and S. Dech, "Influence of image fusion approaches on classification accuracy: a case study," *International Journal of Remote Sensing*, vol. 27, no. 15, pp. 3311–3335, August 2006.
[27] P.J. Curran and J.L. Dungan, "Estimation of signal-to-noise: a new procedure applied to AVIRIS data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 27, no. 5, pp. 620–628, September, 1989.
[28] S. Danaher, E. O'Mongain, "Singular value decomposition in multispectral radiometry," *International Journal of Remote Sensing*, 13(9), 1771-1777, 1992.
[29] I. Davidson and S.S. Ravi, "The complexity of non-hierarchical clustering with instance and cluster level constraints," *Data Mining and Knowledge Discovery*, 14 (2007), 25-61.
[30] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, New York, 1973, 482pp.
[31] W. Eppler, "Canonical analysis for increased classification speed and channel selection," *IEEE Transactions on Geoscience Electronics*, 14, 26-33, 1976.
[32] G. M. Foody, "Thematic map comparison: evaluating the statistical significance of differences in classification accuracy," *Photogrammetric Engineering & Remote Sensing*, 705, 627-633, 2004.
[33] Fortran, *Information technology — Programming languages — Fortran —*, ISO/IEC 1539-1, 1997.
[34] G. Gan, C. Ma, and J. Wu, *Data Clustering: Theory, Algorithms, and Applications*, SIAM, Philadelphia, 2007, 466 pp.
[35] B. Gao, "An operational method for estimating signal to noise ratios from data acquired with imaging spectrometers," *Remote Sensing of Environment*, vol. 43, pp. 23–33, 1993.
[36] B.V. Gnedenko, *Theory of Probability (sixth ed.)*, Gordan and Breach Science Publishers, The Netherlands, 1997, 497pp.
[37] L. Gómez-Chova, L. Bruzzone, G. Gamps-Valls, and J. Calpe-Maravilla, "Semisupervised remote sensing image classification based on clustering and kernel means," in *IGARSS 2008*, 2008.
[38] A. A. Green, M. Berman, P. Switzer, M. D. Craig, "A transformation for ordering multispectral data in terms of image quality with implications for noise removal," *IEEE Transactions on Geoscience and Remote Sensing*, 26(1), 65-74, 1988.
[39] N. Grira, M. Crucianu, N. Boujemaa, "Active semisupervised fuzzy clustering," *Pattern Recognition*, 41(2008), 1834-1844.
[40] M. Halkidi, D. Gunopulos, M. Vazirgiannis, N. Kumar, and C. Domeniconi, "A clustering framework based on subjective and objective validity criteria," *ACM Transactions on Knowledge Discovery from Data*, 1(4), article 18, 25pp., 2008.
[41] J. A. Hartigan, *Clustering Algorithms*, John Wiley & Sons, New York, 1975, 351pp.
[42] S. Hattori, Y. Myint, "Automatic estimation of initial approximations of parameters for bundle adjustment," *Photogrammetric Engineering and Remote Sensing*, 61(7), 909-915, 1995.
[43] HDF5, "HDF5 User's Guide," Hierarchical Data Format (HDF) Group, National Center for Supercomputing Applications (NCSA), University of Illinois at Urbana-Champaign (UIUC), Urbana-Champaign, IL, 2005, 408pp.
[44] K. Y. Huang, "A synergistic automatic clustering technique (SYNERACT) for multispectral image analysis," *Photogrammetric Engineering & Remote Sensing*, 68(1), 33-40, 2002.
[45] G. F. Hughes, "On the mean accuracy of statistical pattern recognizers," *IEEE Transactions on Information Theory*, 14, 55â^63, 1968.
[46] B. Jeon and D.A. Landgrebe, "Partially supervised classification using weighted unsupervised clustering," *IEEE Transactions on Geoscience and Remote Sensing*, 37(2) 1073–1079, 1999.
[47] J. R. Jensen, E. W. Ramsey, H. E. Mackey Jr., E. J. Christensen, and R. R. Shartz, "Inland wetland change detection using aircraft MSS data," *Photogrammetric Engineering & Remote Sensing*, 53(5), 521-529, 1987.
[48] J. R. Jensen, *Introductory Digital Image Processing, A Remote Sensing Perspective*, Prentice Hall, 2005.
[49] H. Jiang, J.R. Strittholt, P.A. Frost, and N.C. Slosser, "The classification of late seral forests in the Pacific Northwest, USA using Landsat ERM+ imagery," *Remote Sensing of Environment*, 91(3–4), 320–331, 2004.
[50] H. Jordan and G. Alaghband, *Fundamentals of Parallel Processing*, Prentice Hall, New Jersy, 2003, 560pp.
[51] K. A. Joseph, R. H. Wynne, J. O Browder, and J. B. Campbell, "Comparison of segment and pixel-based non-parametric land cover classification in the Brazilian Amazon using multitemporal Landsat TM/ETM+ imagery," *Photogrammetric Engineering & Remote Sensing*, in press.

[52] D. Kahaner, C. Moler, and S. Nash, *Numerical Methods and Software*, New Jersey: Prentice Hall, pp. 100–113, 1989.

[53] S. Kaski, J. Sinkkonen, and A. Klami, “Discriminative clustering,” *Neurocomputing*, 69(1–3), 18–41, 2005.

[54] R. J. Kauth, G. S. Thomas, “The tasseled cap—a graphic description of the spectral-temporal development of agricultural crops, as seen by landsat,” in *Proceedings, Symposium on Machine Processing of Remote Sensed Data*, West Lafayette, IN, 41-51, 1976.

[55] M. Kelly, D. Sharri, Q. Guo, and D. Liu, “A comparison of standard and hybrid classifier methods for mapping hardwood mortality areas affected by “sudden oak death”,” *Photogrammetric Engineering & Remote Sensing*, 70(11), 1229-1239, 2004.

[56] M.A. King, P.W. Doherty, R.B. Schwinger, D.A. Jacobs, R.E. Kidder, and T.R. Miller, “Fast count-dependent digital filtering of nuclear medicine images: concise communication,” *Journal of Nuclear Medicine*, vol. 24, no. 11, pp. 1039–1045, 1983.

[57] B. Kulis, S. Basu, I. Dhillon, and R. Mooney, “Semisupervised graph clustering: a kernel approach,” *Machine Learning*, doi:10.1007/s10994-008-5084-4.

[58] A. Kumar, S.K. Ghosh, and V.K. Dadhwal, “Full fuzzy land cover mapping using remote sensing data based on fuzzy c-means and density estimation,” *Canadian Journal of Remote Sensing*, 33(2007), 81-87.

[59] D.A. Landgrebe, and E. Malaret, “Noise in remote-sensing systems: the effect on classification error,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. GE-24, no. 2, pp. 294–300, March, 1986.

[60] Landsat 7 Science Data Users Handbook (May 2006), *NASA. [ONLINE]*, Available: http://landsathandbook.gsfc.nasa.gov/handbook.html.

[61] C. Lawson, *Solving Least Squares Problems*, Prentice Hall, 1974.

[62] D. C. Lay , *Linear Algebra and its Applications*, Addison Wesley, New York, 2000, 486pp.

[63] J.B. Lee, A.S. Woodyatt, and M. Berman, “Enhancement of high spectral resolution remote-sensing data by a noise-adjusted principal components transform,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 28, no. 3, pp. 295–304, May, 1990.

[64] M. Lennon, G. Mercier, and L. Hubert-Moy, “Nonlinear filtering of hyperspectral images and anisotropic diffusion,” in *IGARSS 2002: IEEE International Geoscience and Remote Sensing Symposium and 24th Canadian Symposium on Remote Sensing, Vols I-VI, Proceedings - Remote Sensing: Integrating Our View of the Planet*, pp. 2477–2479, 2002.

[65] G. E. Lowitz, “Stability and dimensionality of Karhunen–Loeve multispectral image expansions,” *Pattern Recognition*, 10, 359-363, 1978.

[66] M. Migliaccio, A. Gambardella, “Microwave radiometer spatial resolution enhancement,” *IEEE Transactions on Geoscience and Remote Sensing*, 43(5), 1159-1169, 2005.

[67] R. F. Musy, R. H. Wynne, C. E. Blinn, J. A. Scrivani, and R. E. McRoberts, “Automated forest area estimation via iterative guided spectral class rejection,” *Photogrammetric Engineering & Remote Sensing*, 72(8),949-960, 2006.

[68] F. Okeke and A. Karnieli, “Methods for fuzzy classification and accuracy assessment of historical aerial photographs for vegetation change analyses. Part I: algorithm development,” *International Journal of Remote Sensing*, 27(2006), 153-176.

[69] W. Pedrycz and J. Waletzky, “Fuzzy clustering with partial supervision,” *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, 27(5), 787–794.

[70] M. Pepe, L. Boschetti, P.A. Brivio, and A. Rampini, “Accuracy benefits of a fuzzy classifier in remote sensing data classification of snow,” in *2007 IEEE International Conference on Fuzzy Systems, VOLS 1–4*, IEEE Electronic Devices Society and Reliability Group, New York, 2007, 492-497.

[71] R. D. Phillips, L. T. Watson, and R. H. Wynne, “Hybrid image classification and parameter selection using a shared memory parallel algorithm,” *Computers & Geosciences*, doi:10.1016/j.cageo.2006.10.014, 2007.

[72] G. Pok, J. Liu, and A.S. Nair, “Selective removal of impulse noise based on homogeneity level information,” *IEEE Transactions on Image Processing*, vol. 12, no. 1, pp. 85–92, January, 2003.

[73] R. L. Powell, N. Matzke, C. De Souza, M. Clark, I. Numata, L. L. Hess, D. A. Roberts, and M. Clark, “Sources of error in accuracy assessment of thematic land cover maps in the Brazilian Amazon,” *Remote Sensing of Environment*, 90, 221-234, 2004.

[74] N.A. Quarmby, “Noise removal for SPOT HRV imagery,” *International Journal of Remote Sensing*, vol. 8, no. 8, pp. 1229–1234, August, 1987.

[75] M.J. Quinn, *Parallel Programming in C with MPI and OpenMP*, McGraw Hill, New York, 2004, 529pp.

[76] G. A. Reams, W. D. Smith, M. H. Hansen, W. A. Bechtold, R. A. Roesch, and G. G. Molsen, "The Forest Inventory and Analysis Sampling Frame," In The Enhanced Forest Inventory and Analysis Program – National Sampling Design and Estimation Procedures, W. A. Bechtold and P. L. Patterson, editors. USDA Forest Service Southern Research Station, General Technical Report SRS-80, pp 11-26, 2005.

[77] J. A. Richards and X. Jia, *Remote Sensing Digital Image Analysis*, Springer, 1999, 363pp.

[78] D.E. Sabol, J.B. Adams, and M.O Smith, "Quantitative subpixel spectral detection of targets in multispectral images," *Journal of Geophysical Research — Planets*, 97(E2), 2659-2672, 1992.

[79] SAS Institute Inc., *SAS OnlineDoc*$^{(R)}$, Version 8, Cary, NC: SAS Institute Inc., 1999.

[80] R.A. Schowengerdt, *Remote Sensing: Models and Methods for Image Processing, second ed.*, San Diego: Academic Press, pp. 127–135, 1997.

[81] M. Seeger, "Learning with labeled and unlabeled data," Inst. Adaptive and Neural Computation, University of Edinburgh, Edinburgh, UK, Technical Report, TR. 2001, 2001.

[82] Z. Sha, Y. Bai, Y. Xie, M. Yu, and L. Zhang, "Using a hybrid fuzzy classifier (HFC) to map typical grassland vegetation in Xilin River Basin, Inner Mongolia, China," *International Journal of Remote Sensing*, 29(2008), 2317-2337.

[83] B.M. Shahshahani and D.A. Landgrebe, "Small sample size problem and mitigating the hughes phenomenon," *IEEE Transactions on Geoscience and Remote Sensing*, 32(5), 1087-1095, 1994.

[84] R. Sivanpillai, C.T. Smith, R. Srinivasan, M.G. Messina, and X. Ben Wu, "Estimating regional forest cover in East Texas using enhanced thematic mapper (ETM plus) data," *Forest Ecology and Management*, 218(1–3), 342–352, 2005.

[85] N. Slonim and N. Tishby, "Agglomerative information bottleneck," in *Proc. of Neural Information Processing Systems Conference*, pp. 617–623, 1999.

[86] G.M. Smith and P.J. Curran, "The signal-to-noise (SNR) required for the estimation of foliar biochemical concentrations," *International Journal of Remote Sensing*, vol. 17, no. 5, pp. 1031–1058, March, 1996.

[87] M. Snir, S. W. Otto, S. Huss-Lederman, D.W. Walker, and J. Dongarra, *MPI: The Complete Reference*, The MIT Press, Cambridge, 1996, 336pp.

[88] S. Still, W. Bialek, and L. Bottou, "Geometric clustering using the information bottleneck method," in *Proc. of Neural Information Processing Systems Conference*, 2003.

[89] G. Strang, *Linear Algebra and Its Applications*, Academic Press, 1976.

[90] N. Tishby, F.C. Pereira, and W. Bialek, "The informational bottleneck method," in *Proc. of the 37th Annual Allerton Conference on Communication, Control and Computing*, pp. 368–377, 1999.

[91] J.A.N Van Aardt and R.H. Wynne, "Examining pine spectral separability using hyperspectral data from an airborne sensor: An extension of field-based results," *International Journal of Remote Sensing*, vol. 28, no. 2, pp. 431–436, 2007.

[92] W. H. A. M. van den Broek, D. Wienke, W. J. Meissen, C. W. A. de Crom, L. Buydens, "Identification of plastics among nonplastics in mixed waste by remote sensing near–infrared imaging spectroscopy. 1. Image improvement and analysis by singular value decomposition," *Analytical Chemistry*, 67, 3753-3759, 1995.

[93] VBMP, *Virginia Base Mapping Program (VBMP) Digital Orthophotography Project*, Virginia Geographic Information Network (VGIN) Advisory Board, Richmond, Virginia, 2003.

[94] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl, "Constrained K-means clustering with background knowledge," in *Proc. of the 18th International Conference on Machine Learning*, Palo Alto, CA: Morgan Kaufmann, 577-584.

[95] J. P. Wayman, R. H. Wynne, J. A. Scrivani, and G. A. Reams, "Landsat TM-based forest area estimation using Iterative Guided Spectral Class Rejection," *Photogrammetric Engineering & Remote Sensing*, 67(10), 1155-1166, 2001.

[96] R. H. Wynne, K. A. Joseph, J. O Browder, P. M. Summers, "Comparing farmer–based and satellite–derived deforestation estimates in the Amazon basin using a hybrid classifier," *International Journal of Remote Sensing*, in press.