3-14-2014

Reconstruction Error and Principal Component Based Anomaly Detection in Hyperspectral imagery

James A. Jablonski

RECONSTRUCTION ERROR AND PRINCIPAL COMPONENT BASED ANOMALY DETECTION IN HYPERSPECTRAL IMAGERY

THESIS

James A. Jablonski, Captain, USA

AFIT-ENS-14-M-11

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENS-14-M-11

RECONSTRUCTION ERROR AND PRINCIPAL COMPONENT BASED ANOMALY DETECTION IN HYPERSPECTRAL IMAGERY

THESIS

Presented to the Faculty

Department of Aeronautics and Astronautics

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Operational Research

James A. Jablonski, BS

Captain, USA

March 2014

AFIT-ENS-14-M-11

RECONSTRUCTION ERROR AND PRINCIPAL COMPONENT BASED ANOMALY DETECTION IN HYPERSPECTRAL IMAGERY

James A. Jablonski, BS

Captain, USA

Approved:

//signed//_____________________________	_6 March 2012_
Kenneth W. Bauer, PhD (Chairman)	Date
//signed//_____________________________	_6 March 2012_
J. O. Miller, PhD (Member)	Date
//signed//_____________________________	_6 March 2012_
David M. Ryer, LtCol, PhD (Member)	Date

AFIT-ENS-14-M-11

Abstract

The rapid expansion of remote sensing and information collection capabilities demands methods to highlight interesting or anomalous patterns within an overabundance of data. This research addresses this issue for hyperspectral imagery (HSI). Two new reconstruction based HSI anomaly detectors are outlined: one using principal component analysis (PCA), and the other a form of non-linear PCA called logistic principal component analysis. Two very effective, yet relatively simple, modifications to the autonomous global anomaly detector are also presented, improving algorithm performance and enabling receiver operating characteristic analysis. A novel technique for HSI anomaly detection dubbed "multiple PCA" is introduced, and found to perform as well or better than existing detectors on HYDICE data while using only linear deterministic methods. Finally, a response surface based optimization is performed on algorithm parameters such as to affect consistent desired algorithm performance.

Acknowledgments

I would like to express my sincere appreciation to my faculty advisor, Dr. Bauer, for his patience, insight, guidance, and encouragement throughout this process. Finally, also wish to thank my thesis committee, Trevor, Todd, and all of my instructors who helped along the way.

James A. Jablonski

Table of Contents

Page

List of Figures

Page

Page

Page

List of Tables

Page

RECONSTRUCTION ERROR AND PRINCIPAL COMPONENT BASED ANOMALY DETECTION IN HYPERSPECTRAL IMAGERY

I. Introduction

1.1 Motivation

An anomaly is something that deviates from what is standard, normal, or expected. The ability to accurately and reliably detect anomalies in real world systems can lead to actionable information. This knowledge can enable better military surveillance, cancer or other health problem diagnosis, as well as prevent information systems network intrusion, credit card fraud, and system or even structural failure (Chandola, Banerjee, & Kumar, 2007) (Chan, Ni, & Ko, 1999). Unfortunately, real-world systems commonly involve high-dimensional multivariate data with many observations. To further complicate matters, the data is often wrought with natural variability and other factors that conceal signal in noise, making anomaly detection difficult (Chandola, Banerjee, & Kumar, 2007).

Airborne or space-based remote sensors offer the ability to survey extremely large land areas quickly and with relatively low cost. Multi-spectral imaging, and subsequently hyperspectral imaging, were developed to yield an efficient increase in classification accuracy from these sensors without an expensive increase in spatial sensor resolution (Landgrebe, 2002). Thus, hyperspectral imagery (HSI) has much potential for surveillance applications, but it also presents many challenges due to natural noise, correlation, sensor variability, spectral mixing, and atmospheric and environmental conditions (Eismann, 2012). The result is a mass of challenging multivariate data, ripe for anomaly detection algorithm development. Although many anomaly detection algorithms are application specific (Chandola, Banerjee, & Kumar, 2007), concepts and techniques developed for one area often prove useful elsewhere. Thus, the intent of this research is to improve current, and explore new, anomaly detection applications for HSI, advancing the overall field of anomaly detection.

1.2 Contributions

In this research, the concept of anomaly detection through residual analysis for HSI data reconstructed after dimensionality reduction is presented. Two new reconstruction-based HSI anomaly detectors are outlined: one using principal component analysis (PCA); and another in the form of non-linear principal components, termed 'logistic PCA.' A very effective, but mathematically simple improvement to the Autonomous Global Anomaly Detector (AutoGAD) algorithm (Johnson, Williams, & Bauer, 2013) is also presented. A novel technique for anomaly detection in HSI dubbed "multiple PCA," is outlined, and found to perform as well or better than existing detectors on HYDICE data. Multiple PCA offers advantages over AutoGAD as its execution time and output are entirely deterministic, while offering advantages over other techniques in that it provides information as to the nature of anomalies discovered. Finally, response surface methodolgy is employed in order to optimize the 'multiple PCA' algorithm.

1.3 Organization

A literature review of a background in HSI, relevant HSI anomaly detection methods, and statistical concepts to be used herein is first presented in Chapter 2. Chapter 3 follows with the presentation of three new anomaly detection techniques for HSI, improvements to the AutoGAD algorithm, as well as algorithm optimization.

Chapter 4 compares the results of the different anomaly techniques and assesses their potential for HSI and other applications. Finally, Chapter 5 presents conclusions and recommendations for further research in the area of HSI.

II. Literature Review

2.1 Chapter Overview

This chapter outlines fundamentals of HSI, a background of common HSI anomaly detection algorithms, and statistical methods for implementing and optimizing HSI anomaly detection. The chapter is thusly divided into five sections: Hyperspectral Imagery, Dimensionality Reduction, Anomaly Detection in HSI, Reconstruction Error Based Anomaly Detection, and Response Surface Methods.

2.2 Hyperspectral Imagery

HSI combines two sensing modalities: imaging and spectrometry (Eismann, 2012). Digital imaging, involves the collection of reflected and/or emitted electromagnetic (EM) radiation intensity levels stored as pixels to scaled x and y positions. Depending on applications, the spatial component may be finely sampled, but with a coarsely sampled EM component. Often pixels estimate an image as the intensity of red, green, and blue corresponding to light response of the human eye (Trussell, 1997). Figure 1 shows the common response curve of the retina, and the wide band of wavelengths common digital imaging discretizes into just three channels.

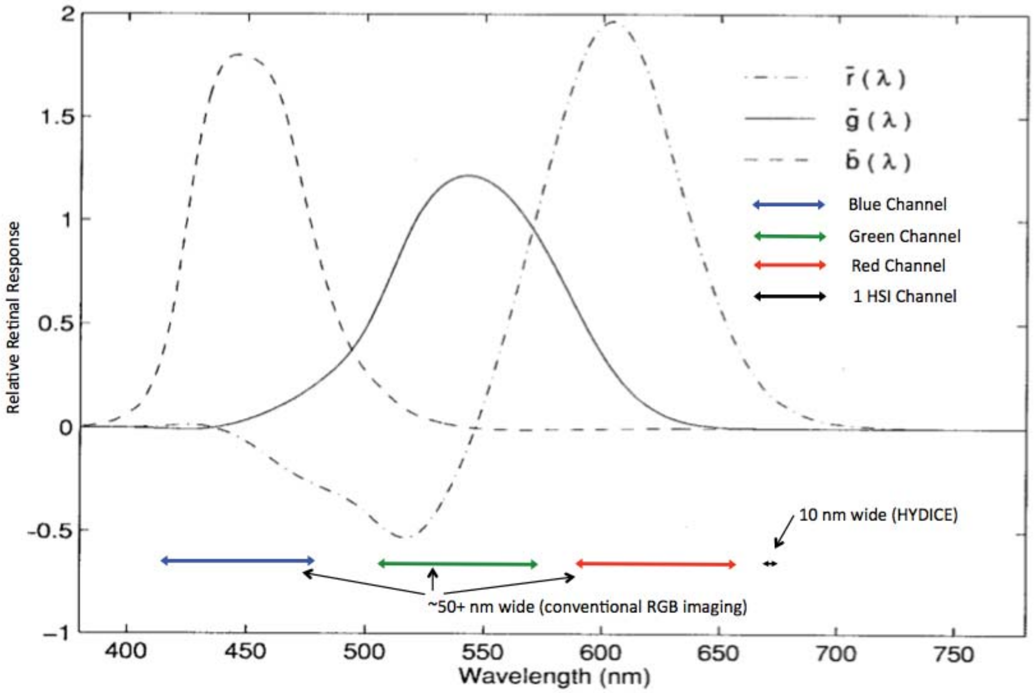


Figure 1: Retinal Response and Digital Imaging Channels (Trussell, 1997)

Spectroscopy, on the other hand, involves using spectrometers to measure a single point of varying EM intensity as wavelength changes over a nearly continuous region of spectrum. Joseph Fraunhofer (1787-1826) invented the first practical spectrometer and used it create an accurate mapping of the visible spectra of the sun (Brand, 1995). Fraunhofer then adopted the same device for use with a telescope, successfully mapping the visible spectra of several stars. Differences in stellar spectra led to Fraunhofer's supposition that stars were materially different from one another. This conjecture proved correct, spectra may be considered the "idioms of atoms and molecules;" molecules "announce their presence" through a series of frequencies emitted or reflected in the electromagnetic spectrum (Brand, 1995). A spectrometer thus collects information associated with the chemical content of a measured substance due to inherent unique quantum molecular properties (Eismann, 2012).

Unlike during Fraunhofer’s time, when he painstakingly recorded spectral bands with mechanical devices, photoelectric detectors first developed during the Second World War, enable digital spectral analysis beyond visible EM radiation with blazing speed (Osborne, Fearn, & Hindle, 1993). Regions of the EM spectrum prove useful for the identification of certain chemical structures, e.g. spectral analysis in the near infrared region (700-2500 nm) is particularly useful for classifying hydrogen bonds due to the nature of molecular vibrations in polyatomic molecules involving hydrogen (Osborne, Fearn, & Hindle, 1993).

In HSI, each pixel is akin to a spectrometer and contains spectral information; an HSI sensor can thus enable both object and material detection as well as classification/identification of a given imaged scene (Eismann, 2012). By definition, HSI are those images containing 20 or more contiguous spectral bands; this differs from multispectral imagery where spectral bands average larger segments of the EM spectrum (Eismann, 2012). The Hyperspectral Digital Imagery Collection Experiment (HYDICE) airborne sensor data used in this research therefore differs from the low-spectral resolution (coarse spectral sampling) illustrated in Figure 1, by covering the spectrum between 400 nm to 2500 nm contiguously with a fine spectral sampling of 10 nm (Nischan, Kerekes, & Baum, 1999).

HSI data is commonly structured as a three dimensional array or “cube,” with the first two dimensions representing the spatial component and the third comprised of the spectrum (Stein, Beaven, Hoff, Winter, Schaum, & Stocker, 2002). For mathematic and matrix manipulation reasons, most image processing algorithms require the data to be transformed into a two-dimensional data matrix in order to perform anomaly detection or classification. The resultant data matrix consists of a $n \times p$ matrix with n pixels each comprised of p spectral bands as depicted in Figure 2.

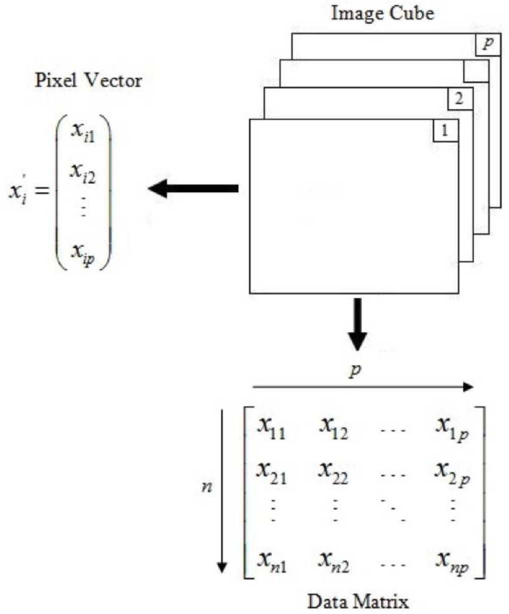


Image Cube
p
2
1
Pixel Vector
$x_i' = \begin{pmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{ip} \end{pmatrix}$
p
n
$\begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{bmatrix}$
Data Matrix

Figure 2: Hyperspectral Data. Reprinted from (Williams J. , 2012)

HSI is currently used in a wide range of applications including: remote sensing, terrain classification, agricultural and environmental monitoring, geological exploration (Stein, Beaven, Hoff, Winter, Schaum, & Stocker, 2002), mine detection (Banerjee, Burlina, & Diehl, Banerjee SVDD, 2006), bathymetry (Sandidge & Holyer, 1998), urban area classification (Bedikstsson, Palmason, & Sveinsson, 2005), drug detection (Rodionova, 2005), law enforcement (Elderding, Thunen, & Woody, 1991), as well as search and rescue applications (Eismann, 2012). The basic concept and method of data collection are presented in Figure 3; HSI imaging systems generally build an image sequentially by capturing spectral pixel information with a one or two dimensional detector array aimed by pointing and stabilization systems. The HYDICE sensor, which collected the data used in this research, is a pushbroom system where images are captured with a linear detector array (one dimensional) oriented perpendicular to the motion of the platform as in Figure 3 (Eismann, 2012). The pushbroom HYDICE sensor differs from whiskbroom sensors, such as AVIRIS, which capture a one-dimensional array oriented parallel to the direction of travel and scanned side-to-side creating a desired image width. Additional HSI methods exist, including step-stare systems, that capture imagery using a two-dimensional array (Eismann, 2012), but these are unexplored herein. Collection methods vary in performance under different pointing schemes and configurations as seen in (Eismann, 2012), and some could lend themselves to on-line HSI anomaly detection as seen in (Bush, 2012).

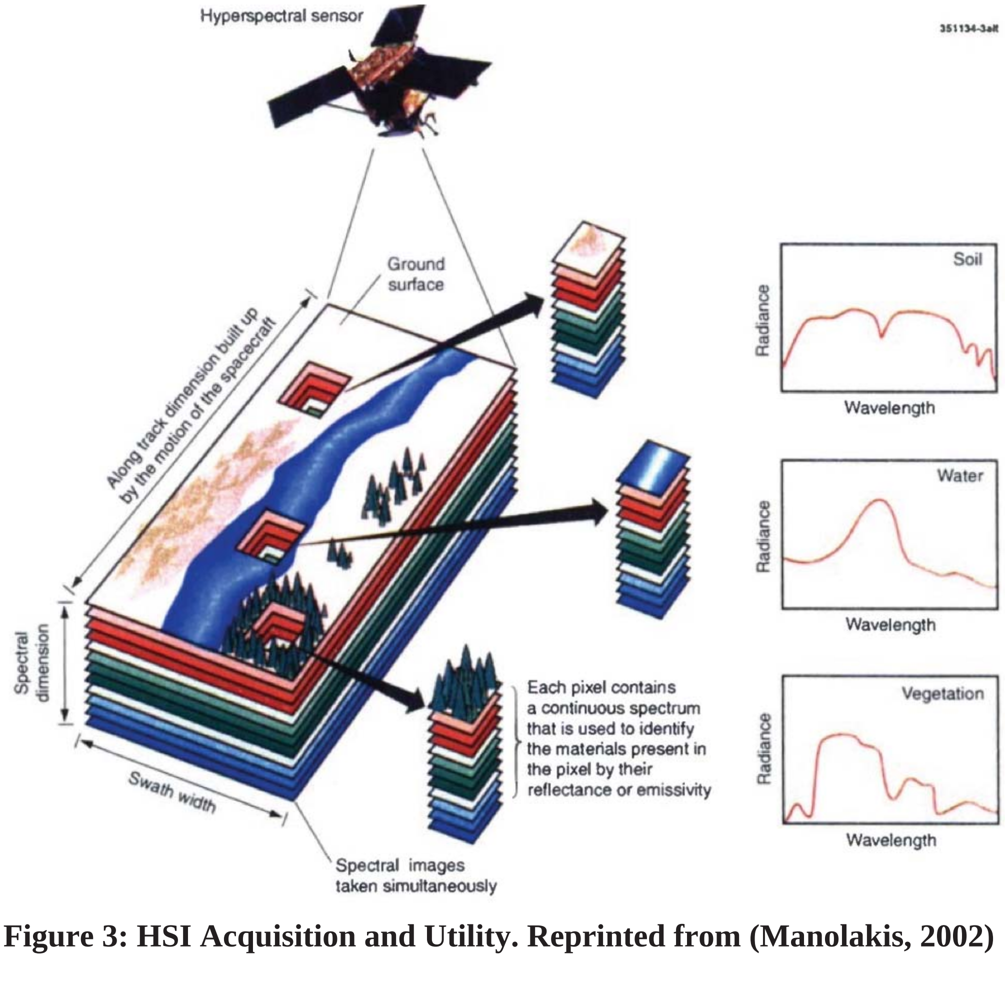

Hyperspectral sensor
351134-3alt
Ground surface
Along track dimension built up by the motion of the spacecraft
Spectral dimension
Swath width
Spectral images taken simultaneously
Each pixel contains a continuous spectrum that is used to identify the materials present in the pixel by their reflectance or emissivity
Soil
Radiance
Wavelength
Water
Radiance
Wavelength
Vegetation
Radiance
Wavelength

Figure 3: HSI Acquisition and Utility. Reprinted from (Manolakis, 2002)

For material recognition, HSI collected radiance is matched against a known spectral signature (radiance), or an atmospherically corrected reflectance signature from a ground experiment. Varying factors such as ground temperature, atmospheric scattering and absorption, solar angle, presence of clouds, shadowing, spectral mixing, and viewing geometry can alter the transmission of radiance and emittance to the HSI sensor (Eismann, 2012). These dynamics must be estimated and compensated for in order to effectively recognize targets using known spectral information.

One area of interest are algorithms that can use HSI data to detect and identify small or sub-pixel (spectrally mixed) objects (Nasrabadi, 2014). In order to compare a ground collected reflectance signal, recognition algorithms require calibration to correct spectral signature differences in images due to in scene conditions; this is accomplished through estimation methods such as in-scene methods, like vegetation normalization, and model based methods, such as MODTRAN (Eismann, 2012). An example of spectral signatures corresponding to surface contents in is shown in Figure 4. Anomaly detection remains of key importance, as it does not require *a priori* information, spectral matching, or the estimation methods mentioned above (Stein, Beaven, Hoff, Winter, Schaum, & Stocker, 2002).

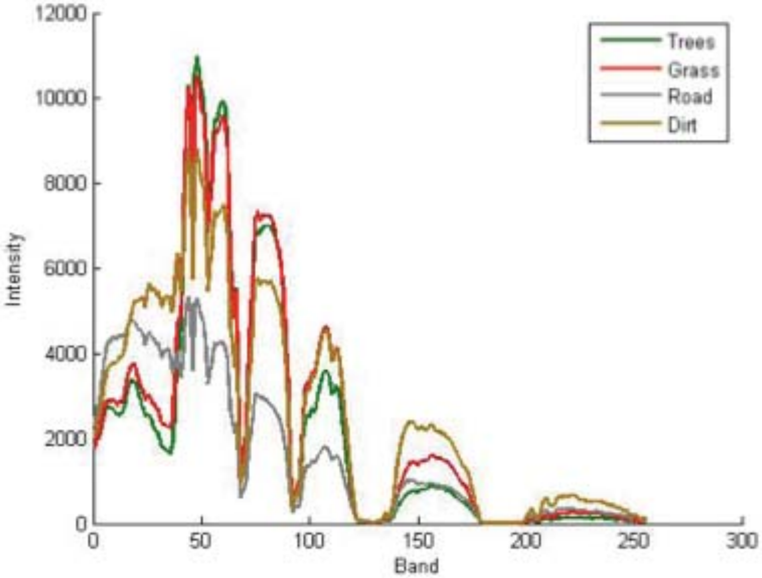


Figure 4: Spectral Signatures. Reprinted from (Smetek, 2007)

Adding another complication, HSI data is highly correlated, both spectrally and spatially (Williams, Bihl, & Bauer, 2013). HSI, like other imagery, increases in spatial correlation as resolution increases with pixel band intensities similar to neighborhood pixels (Ranzato, Krizhevsky, & Hinton, 2010). This inter-pixel correlation brings into question assumptions of normality and independence inherent in many statistical methods used to study larger multivariate problems such as hyperspectral imagery. Furthermore, intra-pixel correlation caused by linear spectral mixing reduces the information content within pixels and thus dimensionality reduction is appropriate (Eismann, 2012).

2.3 Dimensionality Reduction

As with any “big data” problem, the high volume of data inherent within HSI remains demanding on detection algorithms and potentially the storage and transmission capability of devices associated with data collection (Banerjee, Burlina, & Diehl, 2006) (Becker, King, McMullen, & Fahsi, 2013) (Bush, 2012) (Johnson R. J., 2008).

Furthermore, the actual information content of finely sampled bands and spatial pixels is low due to inherent correlation and redundancy. Thus, this larger amount of data often adds more noise than independent pieces of information, and makes algorithms more computationally expensive (Licciardi, Del Frate, Schiavon, & Solimini, 2010). Methods to reduce the dimensionality of HSI are therefore highly desirable and perhaps even necessary to perform hyperspectral data analysis. The primary dimensionality reduction techniques are either feature selection based, where original bands are selected, or feature extraction based, where a transform is used to project the data into a new space. Considered in this research are the feature extraction methods: principal component analysis, logistic principal component analysis, and independent component analysis (ICA).

2.3.1 Principal Component Analysis

Principal components (PCs) are linear combinations of the original variables extracted such that they are uncorrelated and ordered by variance. The first PC accounts for the largest amount of the total variation in the data and the second PC accounts for the second most and so on (Dillon & Goldstein, 1984). Through this concept, PCA achieves dimensionality reduction on a data matrix, $X_{n\times p}$, with n exemplars and p variables, while still explaining an appropriate amount of variation. In practice, PCs are calculated through determination of p eigenvalues, $\lambda_{1\times p} = (\lambda_{(1)}, \lambda_{(2)}, ..., \lambda_{(p)})$, and p eigenvectors, $V_{p\times p} = \left[V_{(1)}, V_{(2)}, ..., V_{(p)}\right]$, of the sample covariance matrix, $S_{p\times p}$, or sample correlation matrix, $R_{p\times p}$, of a data set with original dimensionality p. This is accomplished by solving p simultaneous linear equations:

$$(S - \lambda I)V = 0 \qquad (1)$$

where S is the sample correlation matrix and I is an identity matrix. The resultant eigenvectors are arranged in descending order according the magnitude of their corresponding eigenvalues. The solutions are orthogonal, converted to unit length, and provide a solution to "diagonalize" the original covariance structure thus resulting in the product:

$$V'SV = \Lambda \text{ ; where } \Lambda = \begin{bmatrix} \lambda_{(1)} & & & \\ & \lambda_{(2)} & & \\ & & \ddots & \\ & & & \lambda_{(p)} \end{bmatrix}. \qquad (2)$$

Thus, it can be seen that in this new covariance structure there is no correlation between variables while the total variance in the original structure is retained (Dillon & Goldstein, 1984). The linear transformation described by V is then performed on the original data with the resultant matrix being the full matrix of principal components, T, where

$$T_{n \times p} = X_{n \times p} V_{p \times p} . \qquad (3)$$

The first k components are generally kept such that very little useful information is discarded and the latter principal components are assumed to contain mostly noise (Eismann, 2012). Furthermore, principal components are commonly "whitened" and centered prior to use in algorithms. Whitening refers to scaling all PCs such that they share a variance of 1.0, and centering refers to subtracting the mean vector from each data point such that all variable means are zero (Eismann, 2012). The whitening and centering transformation is shown in (4), where $\Lambda^{-1/2}$ is a matrix where all the diagonal elements are the inverse square root of the eigenvalues, $\mu_{p\times1} = (\mu_{(1)}, \mu_{(2)}, ..., \mu_{(p)})^T$ is a matrix of the feature means, and $\vec{1}_{p\times1}$ is a matrix of ones:

$$Z_{n\times p} = (X - \vec{1}\mu^T)V\Lambda^{-\frac{1}{2}}. \tag{4}$$

Determining the number of components to retain, *k*, involves various heuristics and rules as explored by (Bigley, 2013), (Peres-Neto, Jackson, & Somers, 2003), and (Jackson D. A., 1993). One standard rule of note is Kaiser's criterion where eigenvalues greater than the mean are retained; during PCA on a sample correlation matrix this simplifies to eigenvalues greater than one. This and many other PCA dimensionality assessment techniques are addressed in (Jolliffe, 2002), (Bigley, 2013), (Peres-Neto, Jackson, & Somers, 2003), and (Jackson D. A., 1993).

In 1933, Hotelling (Hotelling, 1933), who coined the term 'principal components,' surmised that there was a smaller "fundamental set of independent variables…which determine the values of the original p variables." Thus the technique was developed to uncover underlying data structure while simultaneously considering the concept of compression and reconstruction, both of which will be explored later in this paper. From an information theory perspective, PCA is the most efficient method of dimensionality reduction due to its accounting for the most variance in a dataset with the least number of dimensions (Christophe, 2011). This makes it a very tempting technique for use in hyperspectral imagery; however, its application in HSI is not without controversy, see (Cherivadat & Bruce, 2003) (Prasad & Bruce, 2008). Despite potential issues, PCAs use on HSI is wide spread, as seen in (Eismann, 2012), (Farrell & Mersereau, 2005), (Shan & Rodarmel, 2002), and (Fountanas, 2004).

In general, the first principal component of HSI data corresponds to broadband intensity variation across the spectra. “The next few capture the primary global spectral differences across the image (Eismann, 2012).” What might be considered anomalies, or the “statistically rare spectral features” dominate the trailing components with low variance. For most HSI or other high dimensional data spaces, a small set of leading principal components is assumed to capture a predominant amount of variance in the data (Eismann, 2012) (Landgrebe, 2002). For example, Figure 5 illustrates the variance structure for one HYDICE image used in this research, ARES1D, with 210 spectral bands. This shape can safely be considered typical and in this case, 99.2% of the variation is explained in just the first 5 out of 145 principal components and over 99.9% in the first 39 (this is not to say that variance explained is always a reliable indicator of the dimensionality of particular data).

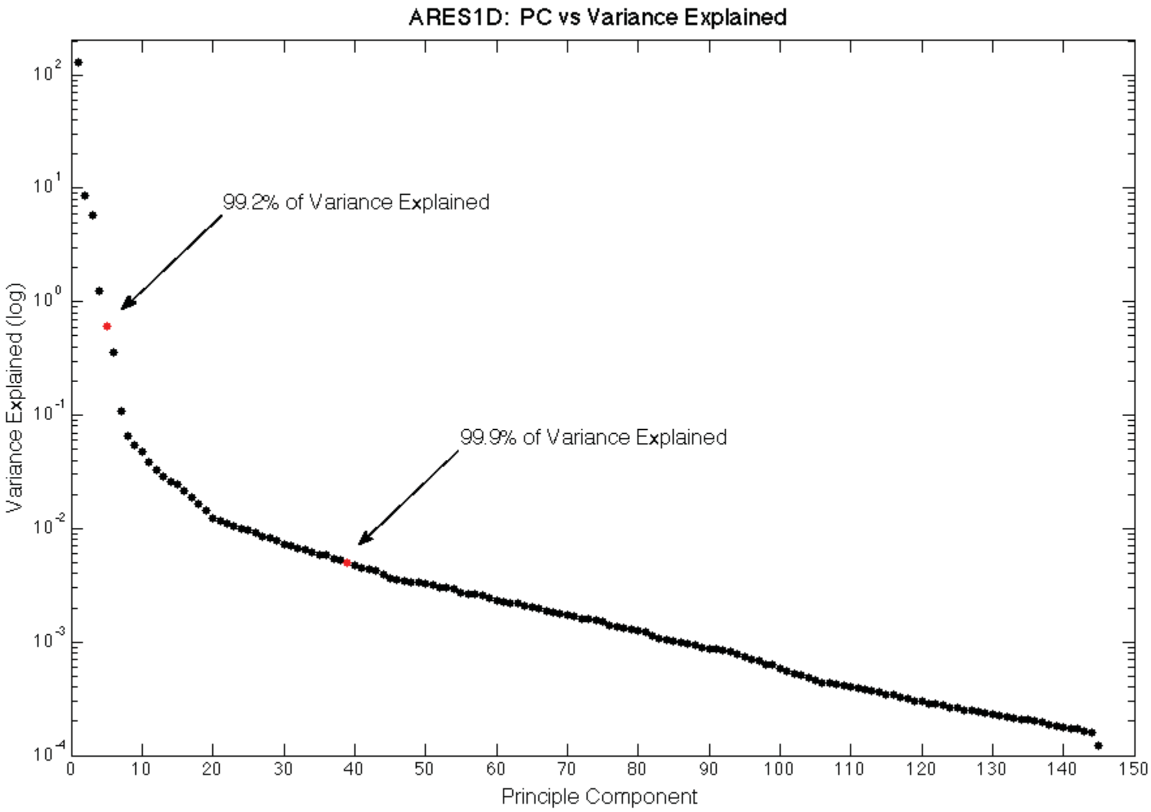


Figure 5: Example of HSI PC Variance Structure

As noted earlier, trailing components may represent rare spectral features; therefore, despite explaining minimal variance they may be important for anomaly or target detection algorithms. Figure 6 contains plots of the first sixteen PCs of ARES1D, an aerial HYDICE image of a desert scene containing six vehicles emplaced on a road vertically through the image. For the purposes of this research, the vehicles are manmade objects of interest (anomalies). As indicated by (Eismann, 2012), the first PC does represent broadband intensity variations; note shadows and contrast due to sunlight originating from the upper side of the image. The following principal components appear to represent other features such as desert vegetation, roads, or perhaps variations in soil composition. Interestingly, by PC 11 what remains appears to be mostly noise, while the vehicles faintly appear again in PC 13 and PC 14.

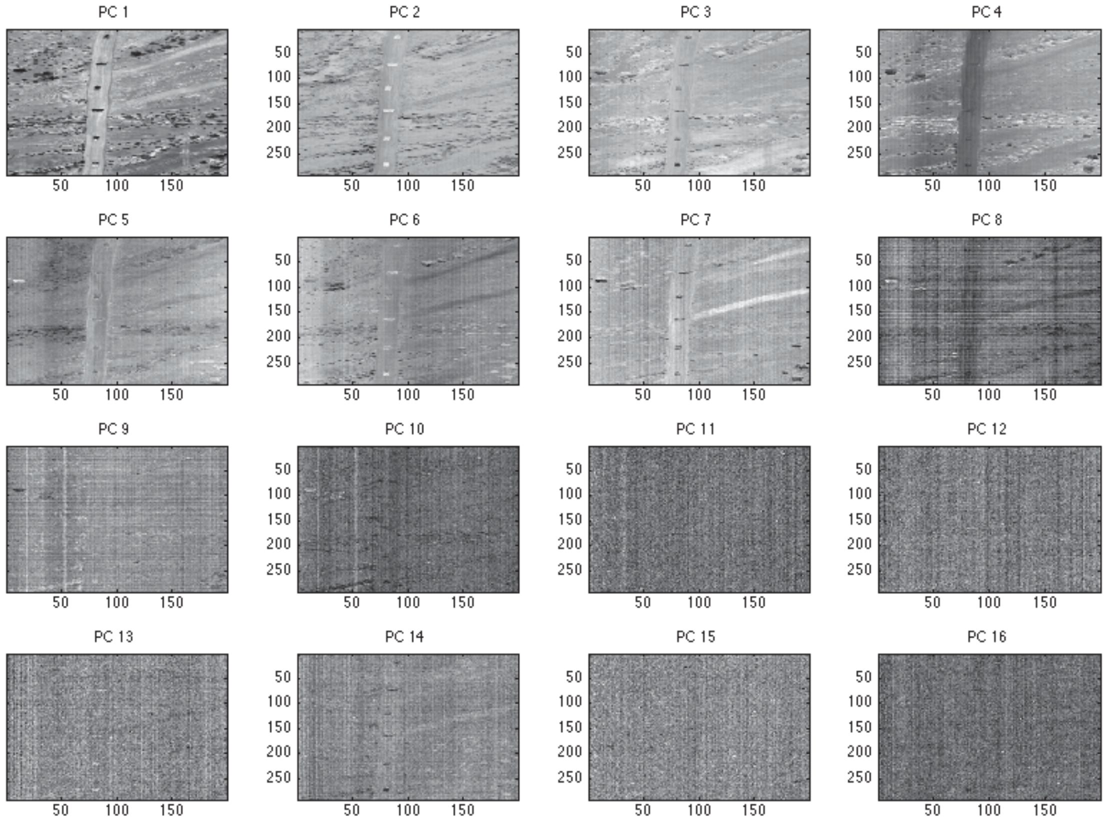


Figure 6: ARES1D, Leading 16 PCs

In this scene the anomalies appear to be very easy to discern from the rest of the scenery using the first few principal components. This is mostly due to contextual cues arising from the linear arrangement and location of the tanks on a road. PC 2 shows the most obvious separation of the anomalies from the scene itself, it seems that all of the manmade objects are geometrically distant from the second principal component's origin. Common spectral characteristics of anomalies in this image are most likely the reason for all targets appearing clearly in the same PC. It is, however, far more common for anomalies to vary in material composition and spectral structure within a scene; these would most likely become discernable in different PCs. In the case of ARES1D, it is interesting to note that in some of the trailing PCs shown in Figure 7 (52 and 58), the man-made anomalies appear again within what appears to be mostly noise. Without shape or contextual cues it is difficult to use these latter PCs to detect outliers or anomalies due to high levels of noise as well as structural artifacts present. For example, the vertical features in Figure 7 may be artifacts due to the motion of the pushbroom sensor, and often eclipse other local features. In PC 52 some of the manmade anomalies become unrecognizable due to a high intensity vertical artifact traversing the road.

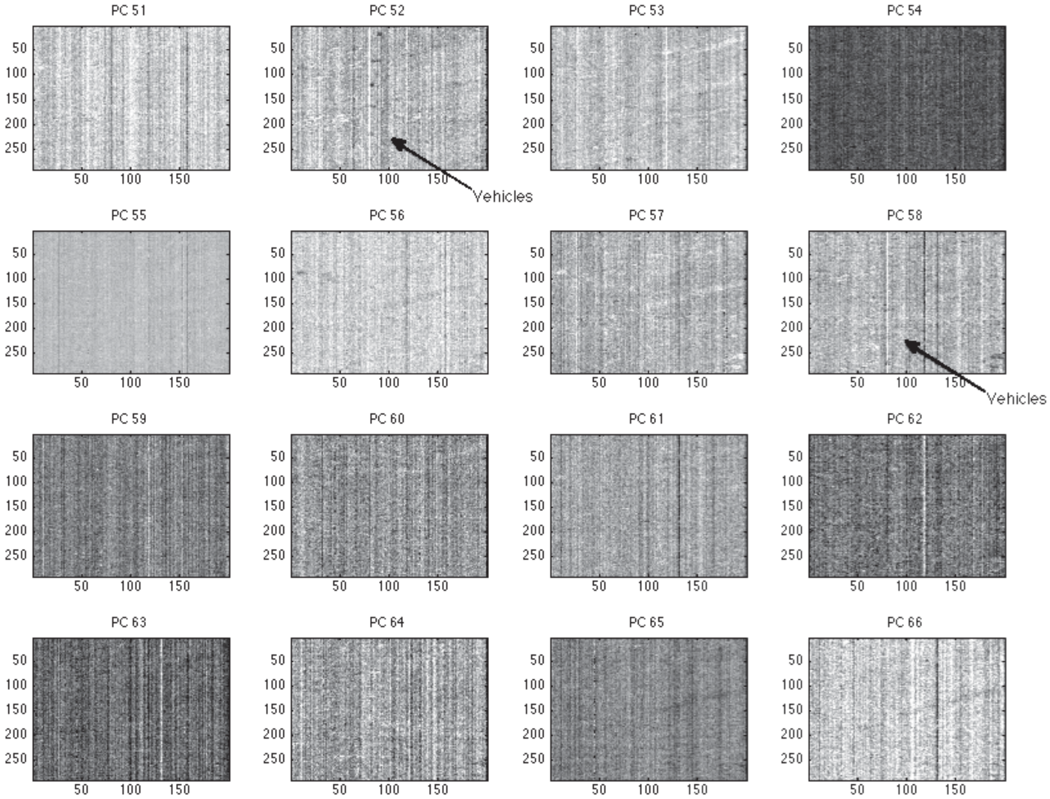


Figure 7: ARES1D, PCs 51-66

2.3.2 Logistic Principal Component Analysis

Thought experiments on gases being microscopic systems interacting classically, as in billiards, led Ludwig Boltzmann to a theoretical explanation of time irreversibility, with the reach of his concepts extending into modern statistical mechanics as the second law of thermodynamics (Flamm, 1983). Here, we find the probability that a given system will be in a specific quantum state, s , of all possible states, S, with energy, ε, at temperature, τ (Kittel & Kroemer, 1980). This probability is derived to be of the form:

$$P(S=s)=\frac{e^{-\varepsilon/\tau}}{Z} \tag{5}$$

The numerator above is known as the Boltzmann factor (Kittel & Kroemer, 1980). The sum of all possible Boltzmann factors for a given system is called the partition function and forms Z (Goldstein, 2002) (Kittel & Kroemer, 1980):

$$Z=\sum_{S} e^{\varepsilon_S/\tau} \tag{6}$$

As one can gather from above, quantum states with low energy are more likely. These systems are dynamic but always “desire” to be in the lowest energy levels as governed by the energy function and ultimately Boltzmann’s concept of entropy (Kittel & Kroemer, 1980). This construct enables dimensionality reduction through Monte-Carlo methods that simulate the activity of a specially structured energy-based system. These systems essentially train a map of the original data features to a smaller number of dimensions using the energy function from above (Hinton, 2010).

2.3.2.1 A Bayesian Connection

It is a useful stretch of logic to equate the denominator in a Bayesian inference problem to the partition function above. Consider the directed acyclic graph (DAG) in

Figure 8 forming a Bayesian belief network. Here, arrows denote causality, one-way dependence between binary random variables (Duda, Hart, & Stork, 2001).

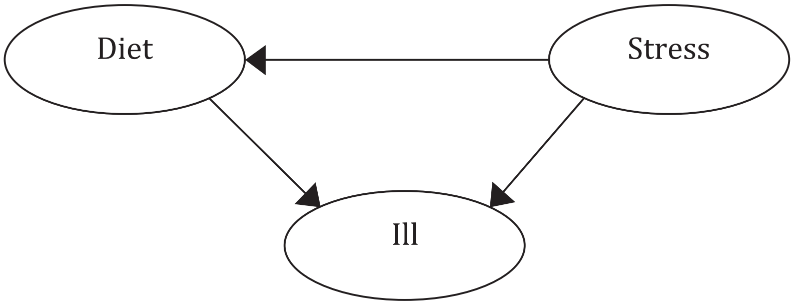


Figure 8: DAG (one way dependence)

The probability that one was stressed given they became ill may be obtained by marginalizing over all combinations of the system where one became ill. This is simply an extension of Bayes' rule, and considering all potential configurations of the random variables in the DAG akin to all possible quantum states in a physical system, a connection to the partition function becomes apparent.

Common techniques in machine learning might really all be boiled down to some Bayesian form. In classification, we try to maximize the intersection of the event that we call an object as class A, and the event that it actually is class A. A problem with most of these models is the implication of one-way or directed causation/dependence. One might ask, "but is this how nature works?" while in reality, "No, there is most likely interdependence."

Further considering the above DAG, if we assume that "your diet and health affect your stress level, and your health affects your diet," then a network of interdependent random variables like this can be modeled as a Markov random field (Kindermann & Snell, 1950). This conceptualization looks similar to a Bayesian Belief Network except there are no arrows denoting one-way dependence, only lines denoting mutual dependence. The example above may be reconsidered as the Markov random field:

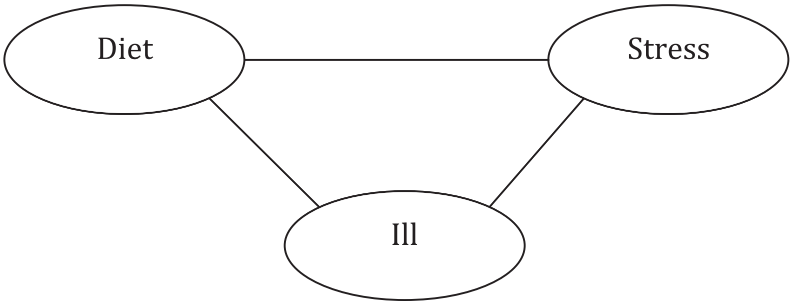


Figure 9: Markov Random Field (two-way dependence)

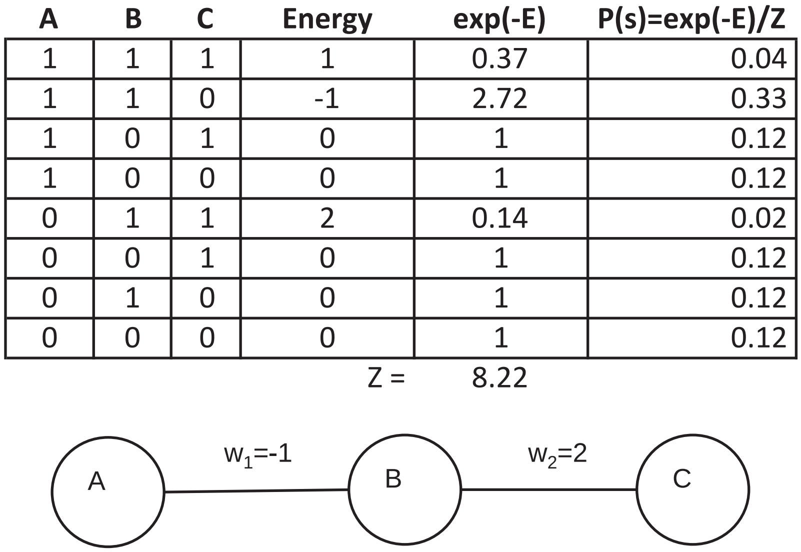
Now, imagine adding another node only connected to stress called “job status” with no connection between the “ill” and “diet” node. The “job status” random variable would be considered statistically independent of the “diet” and “ill” random variable, but would be statistically dependent on the stress variable. Only neighboring or connected variables are dependent. This characteristic is called the local Markov property (Kindermann & Snell, 1950). Further, the system is considered a Markov process in that the probability of a configuration at time *t* only depends on the system’s state at time *t*-1 (Kindermann & Snell, 1950). Hence, the evolution of these systems may be modeled as a Markov chain where the probability of the system moving to a given configuration can be derived from a form of the energy function used in thermodynamics (Kindermann & Snell, 1950).

2.3.2.2 Extending to Boltzmann

A useful formulation using the energy function for effecting dimensionality reduction is,

$$P(S=s)=\frac{1}{Z}e^{-\sum w_{ij}s_j s_i} \tag{7}$$

where, $w_{ij,}$ is simply a positive or negative weight denoting a level of connectedness or dependency between two nodes, and s_i and s_j are the binary states of individual nodes (Duda, Hart, & Stork, 2001). For computational tractability, ensure that $w_{ij} = w_{ji}$. To illustrate, consider the simple Markov random field (MRF) and its corresponding table of energies and probabilities given in Figure 10. The structure consists of three stochastic binary random variable units. Once again, Z is simply the sum of all the Boltzmann factors and low energy states are more likely.



A	B	C	Energy	exp(-E)	P(s)=exp(-E)/Z
1	1	1	1	0.37	0.04
1	1	0	-1	2.72	0.33
1	0	1	0	1	0.12
1	0	0	0	1	0.12
0	1	1	2	0.14	0.02
0	0	1	0	1	0.12
0	1	0	0	1	0.12
0	0	0	0	1	0.12
			Z =	8.22	

Figure 10: Energies and Probabilities for a Simple MRF

2.3.2.3 Restricted Boltzmann Machines

The MRF in Figure 10 is called a Boltzmann machine (Hinton & Sejnowski, 1984). A very useful form of these is the restricted Boltzmann machine (RBM), originally called harmoniums by (Smolensky, 1986). Their formation consists of two layers of weights connecting stochastic binary variables denoting mutual dependence with no intralayer connections; this structure is called a bipartite graph, Figure 11.

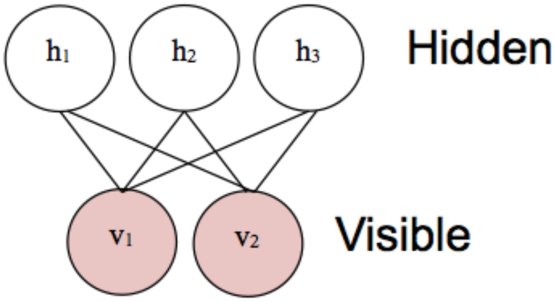


Figure 11: Restricted Boltzmann Machine (RBM)

In RBMs for machine learning there are hidden and visible layers, each consisting of mutually independent stochastic binary random variables. This is exceptionally convenient because if we "clamp" on or set the values of the visible units, we can instantly know the probability of each of the hidden units (Hinton G. , 2007). By computing the effect of a state change for a single hidden or visible unit on the energy of the RBM system, the probability of any given unit being "on" (or 1) is derived to be the sigmoid function:

$$P_{i=1} = \frac{1}{1+e^{-\Delta E_i}} \tag{8}$$

The function in (8) is derived from the Boltzmann distribution, and ensures each random variable unit will likely transition to a state where the change in energy of the system moves the system towards an overall lower energy state (Hinton, 2007) If the system is "unclamped" it will eventually reach "thermal equilibrium" where it hovers around a few likely low energy states, only occasionally existing in any higher energy state (Hinton, 2007).

In order to use a restricted Boltzmann machine to effect dimensionality reduction a visible unit is required for each data variable. An appropriate amount of hidden units is then selected (at this point this is almost arbitrarily chosen depending on the application, situation, and prior working models; a general starting point can be found in (Hinton, 2010)). Training the RBM requires adjusting the weights between units such that if the system, as a Markov chain, was allowed to advance infinitely as in Figure 12, the distribution of realized states of the visible units would resemble that of the input multivariate data (Hinton & Sejnowski, 1984).

The most accurate way of training an RBM is to first clamp on the visible units with an exemplar from our data, and then allow the system to reach thermal equilibrium (this takes one step in the restricted Boltzmann machine) (Hinton, 2007). After this, the visible units are unclamped allowing the system to run freely and move towards thermal equilibrium or low energy states, Figure 12. Weights are then adjusted to make the system more likely to favor outputs like the given exemplar.

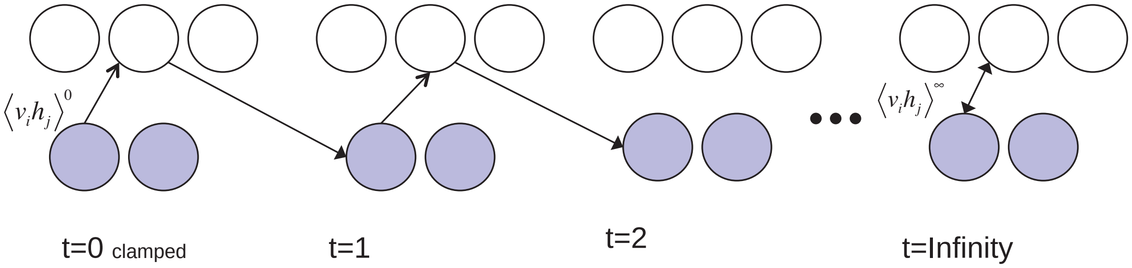


Figure 12: RBM Markov Chain

With the change in the weights between units updated as:

$$\Delta w_{ij} = \varepsilon\left(\langle v_i h_j \rangle_{data} - \langle v_i h_j \rangle_{model}\right) \tag{9}$$

Where ε is an arbitrarily small learning rate (Hinton, 2007).

The energy function generally used in construction of RBMs is slightly different from that presented in (7) as it includes bias terms. It is of the form

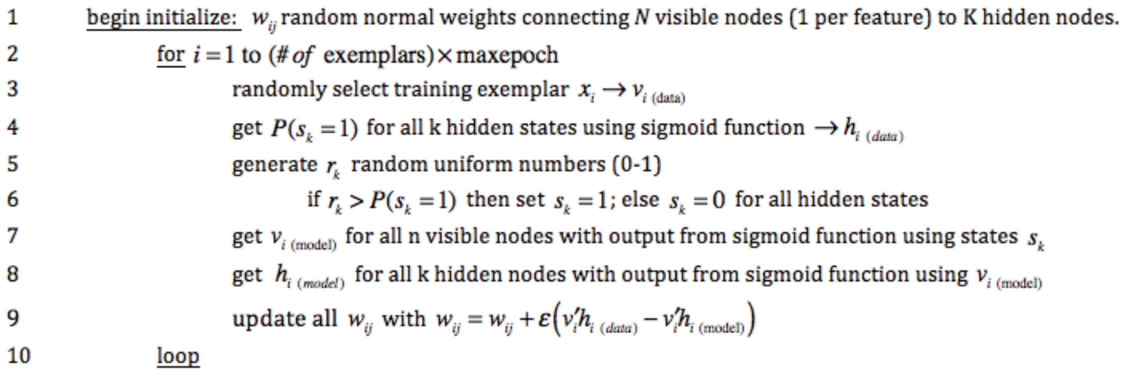
$$E(v,h) = \sum_{i \in visible} a_i v_i - \sum_{j \in hidden} b_j h_j - \sum_{i,j} v_i h_j w_{ij} \tag{10}$$

where, v_i and h_j are the binary states of the hidden and visible units i and j, a_i and b_i are their respective biases, and w_{ij} is the weight between them. The update rules for bias terms is similar to that presented in (9), and are of the form

$$\Delta a_i = \varepsilon_a \left(\left\langle v_i \right\rangle_{data} - \left\langle v_i \right\rangle_{model} \right) \quad \& \quad \Delta b_j = \varepsilon_b \left(\left\langle h_j \right\rangle_{data} - \left\langle v_j \right\rangle_{model} \right) \qquad (11)$$

Where ε_a and ε_b are the visible and hidden unit bias learning rates.

In the above methodology a Monte-Carlo simulation of the Markov chain is run for a very long time and the average output distribution is then compared to our initial distribution (Hinton, 2007). This long simulation is computationally impractical, so Hinton created a less accurate method, albeit proven effective through experimentation, called contrastive divergence (CD) (Hinton, 2007). CD training only requires the Markov chain to move one (or very few) time step(s) and compares the random variable distribution to the distribution at time zero and makes the appropriate weight changes using the same formula as above. Pseudocode for the contrastive divergence technique (excluding bias updates and with only one step) is shown in Figure 13 (Hinton, 2007).



```
begin initialize: $w_{ij}$ random normal weights connecting $N$ visible nodes (1 per feature) to K hidden nodes.
    for $i = 1$ to $(\# of\ \text{exemplars}) \times \text{maxepoch}$
        randomly select training exemplar $x_i \rightarrow v_{i\ (\text{data})}$
        get $P(s_k = 1)$ for all k hidden states using sigmoid function $\rightarrow h_{i\ (data)}$
        generate $r_k$ random uniform numbers (0-1)
            if $r_k > P(s_k = 1)$ then set $s_k = 1$; else $s_k = 0$ for all hidden states
        get $v_{i\ (\text{model})}$ for all n visible nodes with output from sigmoid function using states $s_k$
        get $h_{i\ (model)}$ for all k hidden nodes with output from sigmoid function using $v_{i\ (\text{model})}$
        update all $w_{ij}$ with $w_{ij} = w_{ij} + \varepsilon\left(v_i' h_{i\ (data)} - v_i' h_{i\ (\text{model})}\right)$
    loop
```

Figure 13: Pseudocode for RBM Training

2.3.2.4 Deep Belief Networks and Logistic Principal Components

The result of the described training method is a Boltzmann machine that, when allowed to run freely, tends to transition between states with visible units that correspond to exemplars in a multivariate data distribution. Interestingly, the RBMs are then ‘stacked’ on top of each other. After training the first machine, an additional layer is added consisting of another restricted Boltzmann machine where the visible units are set to the states of the previous layers’ hidden units during training. This is repeated as many times as desired, creating what is called a “deep belief network” (Hinton, 2007).

To result in dimensionality reduction, the final RBM has less hidden units than the original hidden inputs, while the first has more. The final hidden layer probabilities when the first layer is clamped to an exemplar become ‘encoded’ logistic principal components (Hinton, 2007). This model can then be ‘unfolded’ by adding an equal number of opposite layers using transposes of the initial base layers’ weights as in Figure 14. This forms a ‘decoder’ of the lower layers. The outputs of the decoder become probabilistic reconstructions of the original data (Hinton, 2007).

This entire system may now be treated as a normal feed forward neural network. It can be ‘fine-tuned’ using back-propagation to further adjust the weights so that the system more closely models the input distribution. If layer sizes are chosen appropriately (this is essentially more of an art than a science) the decoder outputs are highly accurate reconstructions, or what Hinton calls ‘confabulations’ of the original data (Hinton, 2007). The energy-based model becomes a neural network that ‘understands’ the probability of an image or data vector taking on a certain form. The structure described is illustrated in

Figure 14 for obtaining five logistic principal components from an original data vector with 184 features.

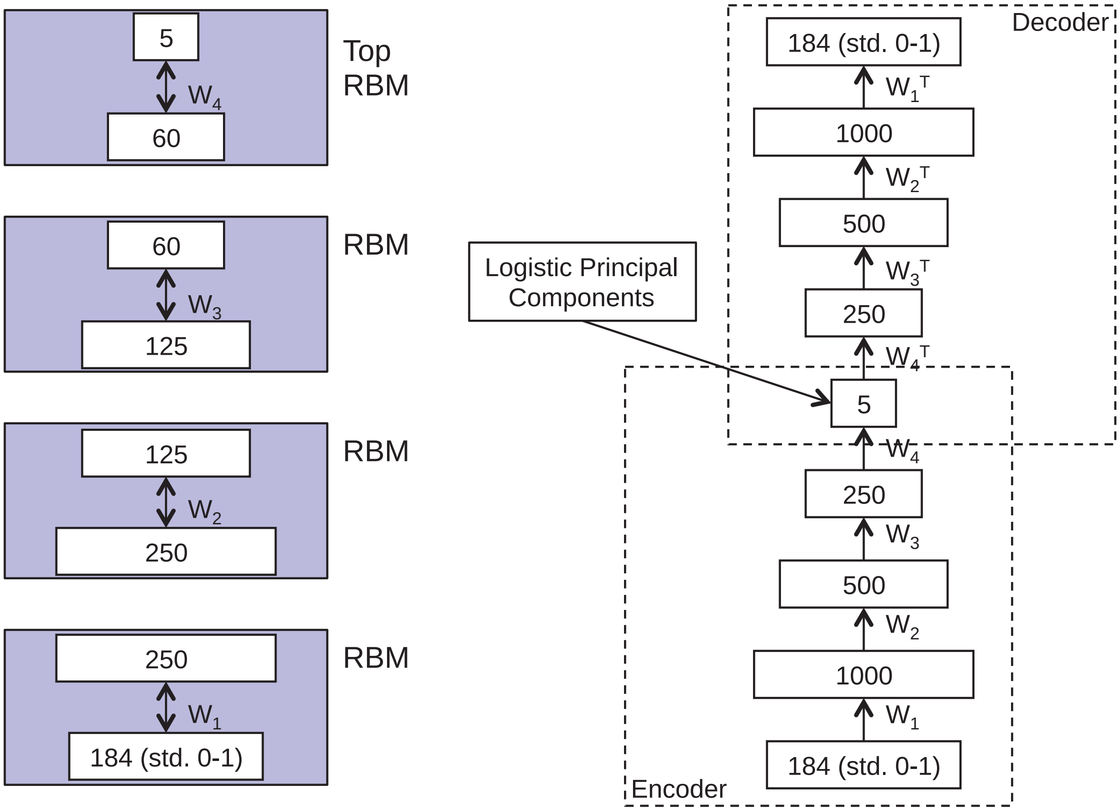


Figure 14: Deep Belief Network

2.3.2.5 Non-Binary Units

Hinton (2010) shows that the previously described form of RBMs works best when the inputs approach binary probabilities, such as do the pixels in the Mixed National Institute of Standards and Technology database (MNIST) of handwritten digits used in (Hinton, 2007). He suggests a form for of the energy function for linear units with independent Gaussian noise for natural images as shown in (12). Here, v_i is the Gaussian state of the visible unit j, h_j is the binary state of the hidden unit j, a_i and b_i

are their respective biases, w_{ij} is the weight between visible and hidden units, and σ_i is the standard deviation of the visible unit noise.

$$E(v,h) = \sum_{i\in vis} \frac{(v_i - a_i)^2}{2\sigma_i^2} - \sum_{j\in hid} b_j h_j - \sum_{i,j} \frac{v_i}{\sigma_i} h_j w_{ij} \qquad (12)$$

The RBMs in this research will convert the hyperspectral data to what can be considered probabilities and not use any Gaussian units as this proved useful for anomaly detection and seemed to provide adequate representations of the data. Gaussian units require orders of magnitude smaller weight updates because the output is not bounded between 0 and 1, as well needing far larger hidden layers (Hinton, 2010). In general, Gaussian RBMs take far longer to train are far less stable than those with binary visible units (Krizhevsky, 2009) (Hinton, 2010). Further, hyperspectral data has been shown to be non-Gaussian (Eismann, 2012), which might further cause instability in the energy function. Many unsuccessful attempts at training non-binary visible units led the author to use the rougher, yet effective approximation to binary inputs for hyperspectral anomaly detection.

2.3.3 Independent Component Analysis

Independent Component Analysis (ICA) seeks to find a set of independent source signals such that each extracted component is statistically independent. This differs from PCA, which extracts a new basis for the data in which all vectors are uncorrelated. ICA is based on a realistic assumption that different physical processes will generate different and independent signals (Stone, 2004). The signals are then collected at a sensor as a linear combination of these independent signals. The technique is commonly used (and theoretically can only be used) when the Gaussian assumption is violated (Eismann, 2012).

In ICA it is assumed that independent source signal vectors, s, are ‘mixed’ together to form a data vector, x, linearly by an unknown mixing matrix A (Stone, 2004). This process is shown in (13) and since the object of ICA is to separate the original signal sources

$$x = As \tag{13}$$

into independent vectors, algorithms in ICA seek to find the inverse of the mixing matrix, A, commonly denoted as, W (Stone, 2004). This matrix divides the data into its source signals as represented in (14). Solving for W is, in general, computationally complex and relies on maximizing total negentropy: the total divergence of the transformed vectors from a Gaussian mixture with the same covariance matrix (Eismann, 2012). An efficient method of ICA called FastICA (Hyvärinen, 1999), finds the components of W using Newton’s method.

$$s = Wx \tag{14}$$

2.3.4 Auto-Associative Neural Networks

Non-linear PCA, an artificial neural network approach, uses neural networks to achieve dimensionality reduction without first applying energy based learning to the model as in (Hinton, 2007). A network architecture with an internal “bottleneck” layer sandwiched between two other hidden layers with inputs mapped to the same outputs was presented in (Kramer, 1991). The bottleneck neurons represent the encoded non-linear principal components. Liccardi et al. (Licciardi, Del Frate, Schiavon, & Solimini, 2010) presented findings for Kramer’s method applied to HSI indicating, “good computational efficiency," and "land cover classification higher than those obtained by some other techniques."

2.3.5 Random Projections

A $p \times k$ matrix with random orthogonal unit length vectors may be used to map multivariate data with p dimensions into a smaller feature space (Kaski, 1998). Ding and Kolacyk (Ding & Kolacyk, 2010) use this concept for privacy reasons, and to reduce the computational complexity of PCA on a large database to effect anomaly detection. For this reason it is considered as a possible augmentation for a PCA reconstruction based anomaly detector for use in hyperspectral data.

2.3.6 Kernel Principal Component Analysis (KPCA)

In Kernel PCA, data is first mapped into a higher dimensional feature space using kernel methods, with the data operated on in the higher dimensional space and then projected back to a lower dimensionality (Nasrabadi & Kwon, 2005). Applying the "kernel trick," and then conducting normal PCA on the resultant features can achieve dimensionality reduction (Bengio, Delalleau, Le Roux, Vincent, Vincent, & Oimet, 2004).

2.4 Anomaly Detection in HSI

Using HSI for remote detection of anomalies and potential objects of interest requires algorithms that can recognize pixels or groups of pixels that exhibit unusual spectral signatures with respect to the global image. The only assumed *a priori* information is that anomalous pixels differ from the global or local background in some manner, that they are significantly less common than the background, and are small in relative size to a physical scene. Due to imaging system considerations and collection altitudes, the ground sampling distance (GSD) of HSI sensors also normally exceeds the anomaly size, resulting in sub-pixel mixing (Eismann, 2012). Further mixing of spectral signatures occurs due to path transmission and physical contaminants occluding targets (Eismann, 2012) (Wong, 2009). In military applications, camouflage, paint, or other techniques may be used to disguise targets as the background, and increase the difficulty of anomaly detection (Eismann, 2012).

The remainder of this section will outline several relevant techniques for hyperspectral anomaly detection; two common preprocessing steps would normally be used in the implementation of all of these methods. First, the data cube must be converted to a $n \times p$ matrix with n pixels each comprised of p spectral bands as depicted in Figure 2. Second, in most HSI data, there are a number of spectral bands that are almost entirely absorbed by the atmosphere due to well-known molecular properties, e.g. water absorption bands, and prohibitively large amounts of noise. The removal of these bands greatly increases the efficiency and effectiveness of most anomaly detection algorithms by increasing the signal to noise ratio inherent in the data.

The anomaly detection performances of various algorithms are compared later in this paper. Methods in this research will be judged according to their true positive fraction (TPF), false positive fraction (FPF), and label accuracy (LA). TPF is the ratio of the total number of anomalous pixels correctly classified to the actual number of existing anomalous pixels. FPF is a similar measure, only it is the number of falsely labeled anomalies divided by the total number of non-anomalous pixels. LA compares the number of anomalous target pixels correctly classified to the total number of pixels classified as anomalies. The formulations of these three fractions cause their ranges to all be [0, 1].

Receiver operating characteristic (ROC) curves will also be used to assess detector performance. The ROC curve is a plot of TPF by FPF as a detector threshold is adjusted with TPF on the vertical axis and FPF on the horizontal (Fawcett, 2006). It is desirable for the curve to track as closely to the top left corner of the plot as possible meaning TPF is almost 1 before FPF climbs significantly above zero. One way to measure this that will be considered later on in this research is the area under the ROC curve (AOC), it is easy to note that the optimal value for the AOC is 1.0. ROC curves can illustrate the classification potential of different techniques, but only assist in finding an actual operating point for a detector threshold. A classification method may have fantastic ROC curves, and yet be of little use if a consistent optimal threshold or operating point across different images is not possible.

2.4.1 Mahalanobis Distance Detector

The Mahalanobis distance detector for HSI uses a test statistic calculated from an images covariance matrix, S, and mean vector of the entire image, $\bar{X}$. The test statistic is formulated where, X_i, is a single data vector or pixel spectrum (Eismann, 2012)

$$r_{MD}(X_i) = (X_i - \bar{X})^T S^{-1} (X_i - \bar{X}) \tag{15}$$

Under the assumption of multivariate normality, r_{MD}, is Chi-Squared distributed. Statistical tests based off of this distribution may then be used to declare anomalies as will be described in the following section. By taking into account covariance, S, the Mahalanobis distance essentially measures the distance from the center of the background distribution while compensating for assumed global multivariate elliptical scattering (Eismann, 2012).

2.4.2 Reed-Xiaoli (RX) Detector

The Reed-Xiaoli detector is a variation of the Mahalanobis distance detector commonly used in HSI anomaly detection. It was developed to better detect anomalies by calculating local rather than global covariance statistics to mitigate problems with spatial correlation and non-stationary statistics (Reed & Yu, 1990). The statistic's calculation is almost identical to that of the Mahalanobis distance except that is calculated locally with the mean vector, $\bar{X}_{local}$, and the covariance matrix, S_{local}, estimated from N_{local} pixels surrounding the test pixel where

$$\bar{X}_{local} = \frac{1}{N_{local}} \sum_{i=1}^{N_{local}} X_i, \tag{16}$$

and

$$S_{local}^{-1} = \frac{1}{N_{local}-1} \sum_{i=1}^{N_{local}} (X_i - \bar{X}_{local})(X_i - \bar{X}_{local})^T. \tag{17}$$

With the resultant formulation for the RX detector

$$r_{RX}(X_i) = (X_i - \bar{X}_{local})^T S_{local}^{-1} (X_i - \bar{X}_{local}). \tag{18}$$

In HSI, the local background consists of a square block of pixels surrounding the pixel under test that effectively moves pixel by pixel through the image as shown in Figure 15. The pixel under test's Mahalanobis distance from the center of the "moving window" of area N, is then compared at a given confidence level, α, to the threshold, χ_α^2, with $N-1$ degrees of freedom. Pixels exceeding this threshold are declared anomalies (Reed & Yu, 1990) (Eismann, 2012). The fact that the test for anomalous data is derived form a generalized likelihood ratio test (GLRT) allows the detector to operate with a constant false alarm rate (CFAR) despite background variation (Reed & Yu, 1990).

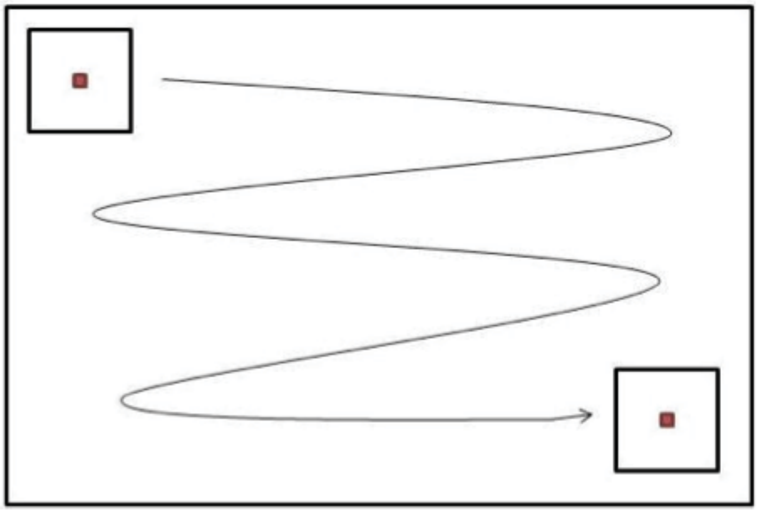


Figure 15: RX Moving Window. Reprinted from (Williams, Bihl, & Bauer, 2013)

The RX algorithm is generally considered the benchmark anomaly detection method for multispectral imagery (Nasrabadi, 2014). The CFAR property also promises a relatively consistent FPF. This does not mean, however, that the method is without problems. The background covariance matrix estimation and its inversion at each pixel demand much computational power. Conducting RX on the leading principal component space (a form of subspace RX or SSRX) of an image is a useful technique to lessen computation burden, but gains speed and cleaner background estimation at the cost of possibly neglecting important information in the discarded principal component directions (Eismann, 2012).

Window size also clearly affects the results of the background estimation and subsequent anomaly classification. False alarms as pixels that would not be called anomalies on a global scale might have high RX scores (Wong, 2009). An example of this would be a tree in the middle of a field. If anomalies are large, they may not appear as anomalies within a small RX window. Variations in background within a given window might even result in the RX algorithm being reduced to an edge detector, creating high FPF rates (Wong, 2009).

Different methods for obtaining better estimations of the local background have been attempted, with geometric methods being the most common. One such method excludes an inner guard band from the background covariance matrix, while another considers an inner window region and outer window region when estimating the same (Nasrabadi, 2014) (Eismann, 2012). Williams et. al. (Williams, Bihl, & Bauer, 2013) present a linear window shape to reduce spatial correlation issues present in all imagery. Finally, an iterative RX method to prevent anomalies from corrupting background estimation was presented by Taitano, Geier, and Bauer (Taitano, Geier, & Bauer, 2010).

2.4.3 Autonomous Global Anomaly Detector (AutoGAD)

While RX considers local statistics for anomaly detection, the autonomous global anomaly detector (AutoGAD), a new approach detects anomalies globally in an image (Johnson, Williams, & Bauer, 2013). AutoGAD is a PCA and ICA-based anomaly detector that advances the work on remote sensing and ICA of (Chiang, Chang, & Ginsber, 2000), (Robila & Varshney, 2002) and (Chen & Zhang, 1999). The algorithm fully automates the process of anomaly detection and generates fast global anomaly declarations with minimal false alarms (Johnson, Williams, & Bauer, 2013).

AutoGAD first reshapes the data and removes absorption bands as outlined in 2.2. Dimensionality reduction is then achieved by PCA. The dimensionality of a hyperspectral image is assessed through a geometric method using the maximum Euclidean distance from the log-scale secant line (MDSL) (Johnson R. J., 2008). The technique estimates the breakpoint between noise and signal by locating the ‘knee’ in the eigenvalue curve (logarithmic scale) and thereby provides a rough solution to methods outlined by (Stocker, Ensafi, & Oliphant, 2003). Essentially a ‘secant’ line is drawn between the first and last eigenvalue in a log scale. The assessed dimensionality, k, corresponds to the eigenvalue with the maximum perpendicular distance from the secant line as shown in Figure 16. In order to prevent errors in dimensionality assessment due to numerical precision problems, eigenvalues less than 10^{-4} are discarded prior to forming the secant line. Johnson showed this technique to be very effective at including enough PCs such that all anomalies are visible in the abundance maps for retained components (Johnson R. J., 2008).

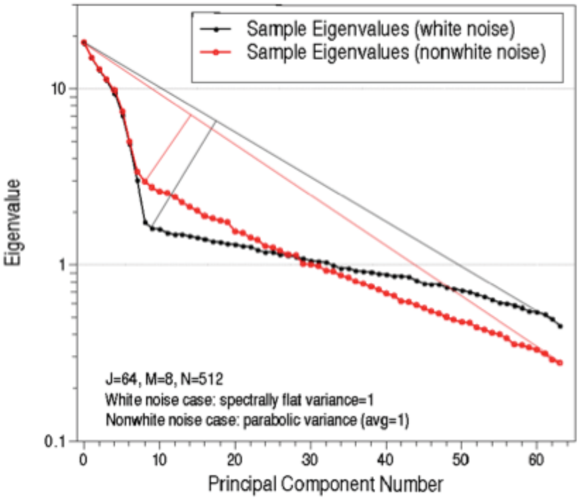


Figure 16: Dimensionality Assessment By Finding the Breakpoint Between Noise and Signal. Reprinted from (Johnson R. J., 2008).

The retained principal components are then whitened, and ICA is performed on the resultant vectors using the FastICA algorithm. Subsequently, the obtained unmixing transformation is applied to the whitened PCs, resulting in k independent components (ICs). Potential anomalies are then nominated using a zero-detection histogram method described by (Chiang, Chang, & Ginsber, 2000). The zero-detection algorithm first constructs histograms of scores for each IC. The location, ϑ, of the first histogram bin with frequency of zero is then identified in each IC score histogram. Pixels associated with scores greater than ϑ are considered anomalous as in (19). A graphical example is offered in Figure 17.

$$\left(Score_i > \vartheta_{\text{Location of First Empty Histogram Bin}}\right) \rightarrow X_i \in anomalies \tag{19}$$

This method is, of course, very sensitive to the bin width, ω, chosen during histogram construction. Wider bins will reduce the sensitivity of the detector and narrow bins will increase the sensitivity and result in more false positives (Johnson R. J., 2008).

The AutoGAD algorithm then proceeds to identify ICs with high signal power by dividing the previously identified potential target variability by the variability of the background. A measure of signal to noise ratio (SNR) is created with the aforementioned fraction. Large area classes such as road and different terrain types generally have low SNR compared to true anomalies. An appropriate threshold is used such that ICs with low SNRs are then discarded in the final anomaly declaration (Johnson, Williams, & Bauer, 2013).

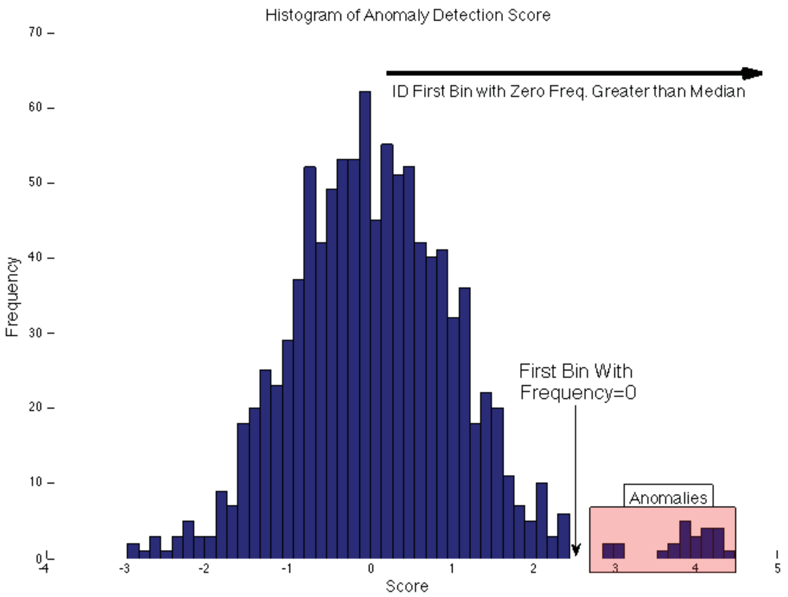
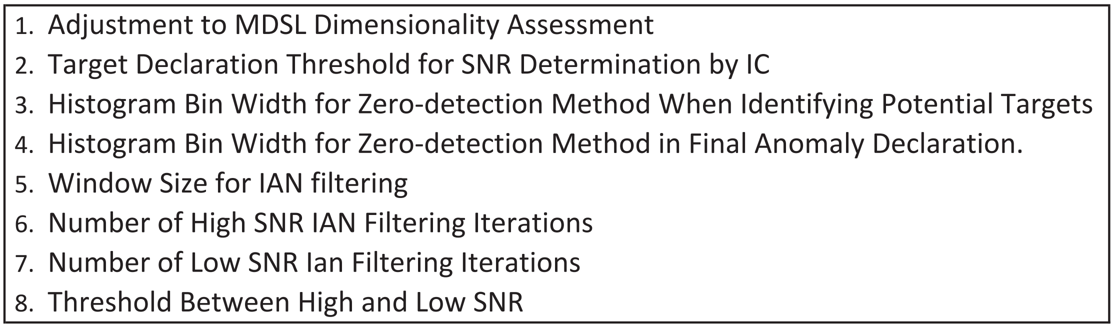


Figure 17: Zero-detection Method

The ICs remaining are then filtered to reduce noise and false positives (Johnson R. J., 2008). Iterative adaptive noise filtering (IAN) is used as it filters more heavily in areas where the variance is close to system noise while not greatly filtering areas with significant signal (Johnson R. J., 2008). An appropriate level of filtering iterations is selected based off of thresholds in SNR. Higher SNR signals are filtered less while lower strength signals are filtered more to reduce background noise. False positives due to noise are thus reduced, while true positives from signal are generally untouched.

Lastly, the zero-detection histogram method is implemented on the remaining filtered ICs resulting in the algorithms final anomaly declaration. Results with this algorithm are promising, but require detailed calibration for scene and sensor types (Johnson R. J., 2008). Figure 18 lists all the parameters and thresholds that need to be optimized for countless different environmental and sensor conditions. Further, because the algorithm uses FastICA (Johnson R. J., 2008) (a process that starts with a random seed), inconsistent results and processing times are possible when running the algorithm on a single image.



1. Adjustment to MDSL Dimensionality Assessment
2. Target Declaration Threshold for SNR Determination by IC
3. Histogram Bin Width for Zero-detection Method When Identifying Potential Targets
4. Histogram Bin Width for Zero-detection Method in Final Anomaly Declaration.
5. Window Size for IAN filtering
6. Number of High SNR IAN Filtering Iterations
7. Number of Low SNR Ian Filtering Iterations
8. Threshold Between High and Low SNR

Figure 18: AutoGAD Algorithm Parameters

2.4.4 Support Vector Data Description

Another different approach to HSI anomaly detection is the support vector data description (SVDD) anomaly detection algorithm for HSI which utilizes a kernel method for modeling the support of a distribution (Banerjee, Burlina, & Diehl, Banerjee SVDD, 2006). It also relies comparing an exemplar to neighborhood pixels and thus is similar to the RX algorithm. The non-parametric model used in SVDD does not rely on the multivariate normal assumption, as does RX. This proposes a potential advantage over RX and many other algorithms and has shown very competitive results (Banerjee, Burlina, & Diehl, Banerjee SVDD, 2006). Results of the SVDD algorithm will thus be used for performance comparisons during chapter three of this research.

2.5 Reconstruction Error Anomaly Detection Methods

A reconstruction error may be obtained when the results of a dimensionality reduction are re-projected into the data’s original space. A linear and non-linear PCA based reconstruction error anomaly detection method are outlined and discussed in this section. Methods such as these can achieve compression and data analysis via dimensionality reduction as well as perform anomaly detection. Thus, highly efficient reconstruction methods might result in greater data storage and computational efficiency if implemented with HSI.

2.5.1 PCA Subspace Method

PCA may be used to "compress" random multivariate normal data $X_{n\times p} \in \Re^{p}$. The first k PCs of the data yield a prediction of X through $\hat{X}_{n\times p} = X_{n\times p} V_{p\times k} V_{p\times k}^{T}$, and this prediction is essentially a re-projection of the data into the original feature space. Using the residuals of this prediction as a statistic to detect multivariate outliers was introduced in 1957 by (Jackson & Morris, 1957). Interestingly, PCA reduction and the residual technique mentioned were suggested due to high correlation within the 'photographic processing' data set studied by Jackson and Morris (Jackson & Morris, 1957), and as mentioned previously, similar problems exist within HSI. Later, Jackson and Mudholkar (Jackson & Mudholkar, 1979) further defined the value for testing goodness of fit and multivariate quality control with the statistic

$$Q_i = (X_{i(1\times p)} - \hat{X}_{i(1\times p)})(X_{i(1\times p)} - \hat{X}_{i(1\times p)})^{T} \tag{20}$$

where, under the assumption of multivariate normality, Q is a linear combination of i.i.d. chi-square random variables. The resultant distribution, when k principal components are retained, is,

$$Q \sim \sum_{i=k+1}^{p} \lambda_i Z_i^2 \tag{21}$$

where λ_i are the eigenvalues of the sample covariance matrix, and Z_i are i.i.d. standard normal random variables. In (Jackson & Mudholkar, 1979) an approximation to the normal distribution is presented based on a power transformation, but as presented by Ding and Kolacyk in (Ding & Kolacyk, 2010), the distribution of Q can approximated as normal with the following mean and variance.

$$\mu = \sum_{i=k+1}^{p} \lambda_i \qquad \sigma^2 = 2\sum_{i=k+1}^{p} \lambda_i^2 \tag{22}$$

This approximation is justified by the central limit theorem, and robust to departures from normality as the number of discarded components increases. Exemplars exceeding a given threshold of reconstructive error using probabilities from the normal distribution function will thus be considered anomalies or outliers in this method.

2.5.2 Replicator Neural Network Anomaly Detection

These methods utilize auto-associative neural networks of many forms to provide a reconstructive error anomaly score. A multi-layer feed forward neural network is constructed and trained such that it has the same number of output and input neurons (Chandola, Banerjee, & Kumar, 2007). The testing phase would evaluates each exemplar, x_i, using the trained network to obtain a reconstruction, o_i. A reconstruction error, δ_i, is then obtained for each data point by summing over *p* features for each data point as in (19). The reconstruction error is then used as an anomaly score for each test instance. A useful list of replicator neural network applications to anomaly detection may be found in (Chandola, Banerjee, & Kumar, 2007).

$$\delta_i = \frac{1}{p}\sum_{j=1}^{p} (x_{ij} - o_{ij})^2 \tag{23}$$

2.6 Response Surfaces and Robust Parameter Design

Target detection in HSI demands the development of classification methods that operate consistently despite embedded random effects such as solar and viewing angle, scale, noise, background, and GSD issues. Assessing anomalies detected within a scene expends valuable man-hours, while failure to cue to anomalies might result in missed targets. Operating parameters and thresholds for HSI target detection algorithms, therefore, should be set properly to effect optimal and consistent results across varying images.

Response surface methodology (RSM) offers a method to accomplish this by empirically modeling an algorithms output with a regression model (Myers, Montgomery, & Anderson-Cook, 2009). A response surface model for algorithm optimization takes the form

$$y = f(\xi_1, \xi_2, ..., \xi_k) + \varepsilon, \tag{24}$$

where the function *f* models an output of the algorithm, *y*, the predictor variables, ξ, are the algorithm parameters, and the error term ε represents the sources of variability in the algorithm not accounted for in *f* (Myers, Montgomery, & Anderson-Cook, 2009). The errors are assumed gaussian with a mean of zero. Predictor variables are often converted from their actual values to coded variables, *x*, to facilitate experimental design and the response function is written

$$f(x_1, x_2, ..., x_k) + \varepsilon, \tag{25}$$

The response surface models considered in this research will be second order polynomials. The models are thusly written

$$y = \beta_0 + x'\beta_1 + x'\beta_2 x + \varepsilon, \tag{26}$$

where β_0 is the intercept, x is a vector of parameter settings, β_1 is vector of control variable coefficients, β_2 is a matrix containing the quadratic control variable coefficients, and ε represents the error. The range of the algorithm parameters before coding and experimental design must be limited such that the response can be accurately modeled by the second order function (Myers, Montgomery, & Anderson-Cook, 2009). A factorial experiment is conducted to generate a sample of the output *y* for the selected design space. Parameters in the response model may then be obtained by the method of least squares, and the final function is optimized within the design space to obtain the optimal settings.

Robust parameter design offers a method to modeling an algorithms output and variability while also considering uncontrollable noise factors (Montgomery, 2009). RPD was developed by Genichi Taguchi as an experimental design approach to optimizing and reducing variability in product output from physical processes (Myers & Montgomery, 2002). The approach has been extended and adapted since and can be used in many areas. RPD may be performed as long as there is at least one interaction between a control variable and an uncontrolled noise factor (Mindrup, 2011). Uncontrolled noise in HSI is inherent in each image due to various factors enumerated earlier, and control variables are the parameters set for different detection algorithms.

The RPD and RSM considered in this research will utilize a form of the dual response surface optimization approach developed by Lin and Tu (Lin & Tu, 1995). Lin and Tu provide a single number is best criterion for optimization after the construction of a response surface model (RSM) of a given system. Typically, RSMs for robust parameter design utilize second order models and ignore higher order interactions due to the sparcity of effects (Mindrup, 2011). The general matrix form for of the response model becomes:

$$y\left(x,z\right)=\beta_0+x'\beta_1+x'\beta_2x+z'\gamma+x'\Delta z+\varepsilon \tag{27}$$

where β_0 is the intercept, x is a vector of parameter settings, β_1 is vector of control variable coefficients, β_2 is a matrix containing the quadratic control variable coefficients, z is a vector of noise variables, γ is a vector of noise variable coefficients, and Δ is a matrix of noise by control variable coefficients. The model is then split into mean response and variance models (Montgomery, 2009).

The Lin and Tu criterion minimizes deviation or mean squared error from a desired target response for a response surface model (Lin & Tu, 1995). The Lin and Tu criterion may be written

$$MSE_T=(\hat{\mu}_y-T)^2-\hat{\sigma}_y^2 \tag{28}$$

Many modifications of the Lin and Tu method have been suggested including goal programming to further specify output characteristics, and target variance outputs (Mindrup, 2011).

III. Methodology

3.1 Chapter Overview

This chapter begins with a description of the hyperspectral data used in this research. In order to facilitate the development and exploration of algorithms presented in this chapter two contrasting images are highlighted. The chapter then outlines an improvement to the AutoGAD algorithm and two new reconstruction error based hyperspectral anomaly detection techniques. Finally, it will conclude with a presentation and subsequent optimization of an anomaly detector dubbed “multiple PCA,” applied to HSI.

3.2 Hyperspectral Data

Data used in this research is from the Forest I and Desert II Radiance collections of the Hyperspectral Digital Imagery Collection Experiment (HYDICE), as discussed in 2.2. Images contain 210 bands of radiance data with 10nm spectral resolution ranging from 400-2500nm in wavelength. Unless otherwise noted, all images were captured with a pushbroom sensor. Unless otherwise noted, they were collected at approximately 5000 ft. above ground level (AGL).   Two of the sample images, one from the forest collection, ARES1F, and one from the desert collection, ARES1D will be used to demonstrate algorithm performance throughout the rest of the chapter.

Visible color representations of ARES1D and ARES1F are shown in Figure 19 along with their corresponding target masks.   Targets are shown in gray with fringe pixels (those containing a mix of both background and target spectral signature) shown in white. The targets in ARES1F consist of vehicles and other smaller objects arranged linearly on the left with tarps and camouflage netting covering larger objects on the right.

ARES1D consists of only a single row of vehicles along a road traversing vertically through the image.   The images were chosen as a focus in algorithm development because of two contrasting elements besides the obvious environmental difference (Forest vs. Desert). First, target pixels cover 3.3% of the image area in ARES1F as opposed to just over 0.41% in ARES1D. The forest image also has some much larger anomalies. Additonally, this research noticed considerable levels of image noise and artifacts present in ARES1D throughout experimentation that seem to cause problems for many algorithms, whereas ARES1F exhibited little of the same.

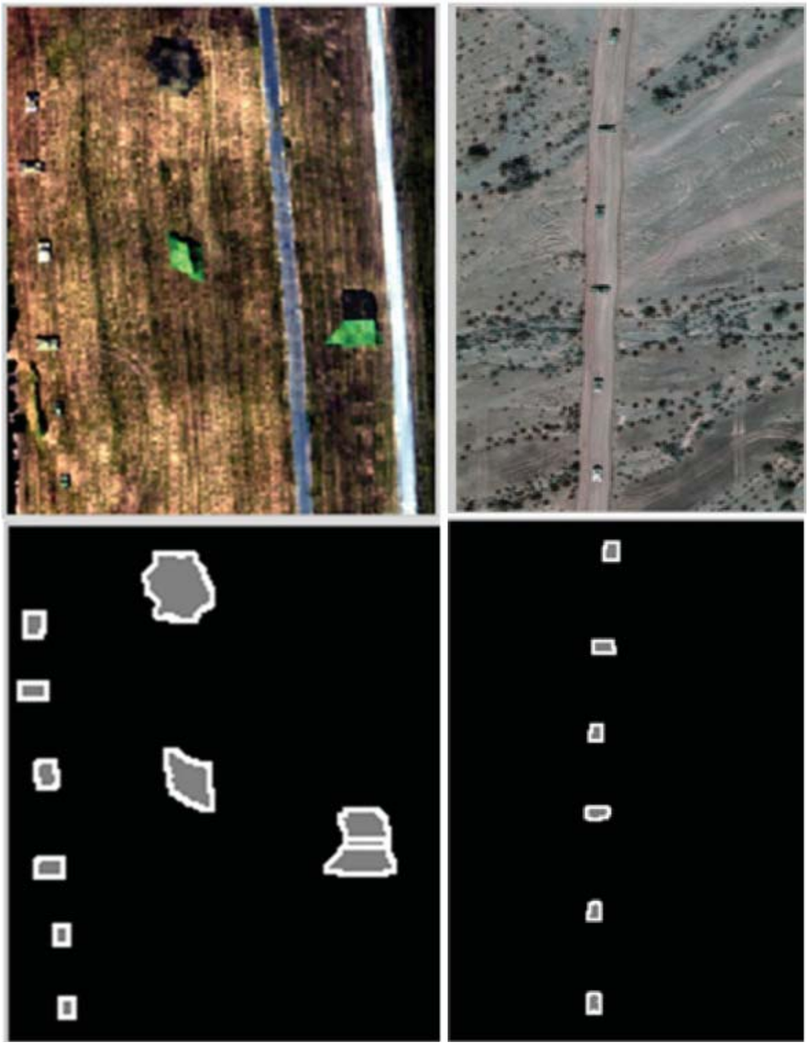


Figure 19: Contrasting Data Sets

3.3 AutoGAD Improvements

The AutoGAD algorithm presented by Johnson (2008) had very positive anomaly detection results on many of the images used in this research; however, it underperformed on some images and there was not yet a straightforward way to generate ROC curves. The algorithm was designed to be fully autonomous, and so the algorithm parameters needed to be set to work effectively across a large range of images. The results presented in (Johnson, Williams, & Bauer, 2013) are shown in Table 1, and illustrate the underperformance on TPF in some images, e.g. ARES4F, ARES1D, and ARES2F. FPF seemed to remain consistently low, but the inconsistency in TPF was troubling. Unfortunately, it was also difficult to visualize the ROC of the algorithm and diagnose the problem.

Table 1: Original AutoGAD Outputs

	Pixels	Mean TPF	Mean FPF
ARES1F	**30560**	0.9571	0.0017
ARES2F	**47424**	0.6599	0.0027
ARES4F	**16400**	0.461	0.0036
ARES1D	**57909**	0.8298	0.0034
ARES3F	**30736**	0.8723	0.0025
ARES2D	**22360**	0.8845	0.0003
Mean	**34232**	**0.777**	**0.002**

Johnson (2008) showed that a key parameter in the AutoGAD algorithm was the bin width, ω, chosen when using the zero-detection histogram method. Larger bin widths would result in a less sensitive detector with lower TPF and FPF rates, while smaller bins would result in increased sensitivity. Generating a ROC curve was therefore theoretically possible by varying ω over an appropriate range. Unfortunately, this was very processor intensive, as it required computationally demanding repetitive sorting over all retained independent components. Furthermore, the algorithm is fairly robust to the bin-width parameter, making the problem less tractable.

Therefore, the first proposed improvement to the AutoGAD algorithm is a simple method to visualize its receiver operating characteristics. Generating a ROC curve necessitated moving the autonomously designated threshold for anomaly declaration for each independent component simultaneously. This could be accomplished by multiplying all independent component's zero-bin detection thresholds, ϑ , by nominal factor, F , and re-computing anomaly declarations. This results in the slight modification to (19),

$$\left(Score_i > F \cdot \vartheta_{\text{Location of First Empty Histogram Bin}}\right) \rightarrow X_i \in anomalies \tag{29}$$

Thus, varying F over an appropriate range while repeatedly re-computing TPF and FPF resulted in a usable ROC curve. The range for F was chosen nominally as [-10,10]. One can see the effective range of the detection threshold in Figure 20 below of 1.566 after this process would be [-15.66, 15.66] (IC scores can be negative).

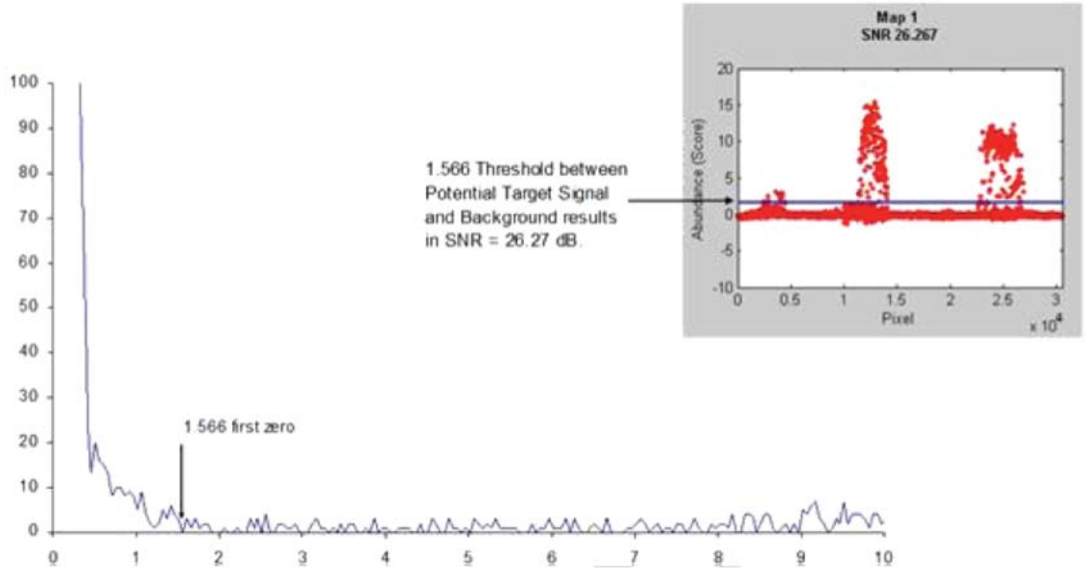


Figure 20: Zero-Bin Method. Reprinted From (Johnson, Williams, & Bauer, 2013)

After studying ROC curves generated in the manner described above, it became apparent that the AutoGAD algorithm was probably not functioning optimally. This can be noted in Figure 21 showing ROC curves for the all of the images with results displayed in Table 1, where the square markings along each curve indicate the original algorithm operating points (note that the axes do not range from [0,1]).   Clearly, and most dramatically in ARES4F and ARES2F, the algorithm had the potential to operate with much higher TPFs while maintaining comparable FPFs.   It was noted that the underperformance was, at least loosely, correlated with deviations from the average image size. This is apparent in the results above where the two largest images ARES1D and ARES2F, as well as the smallest image ARES4F have the three lowest TPF scores.

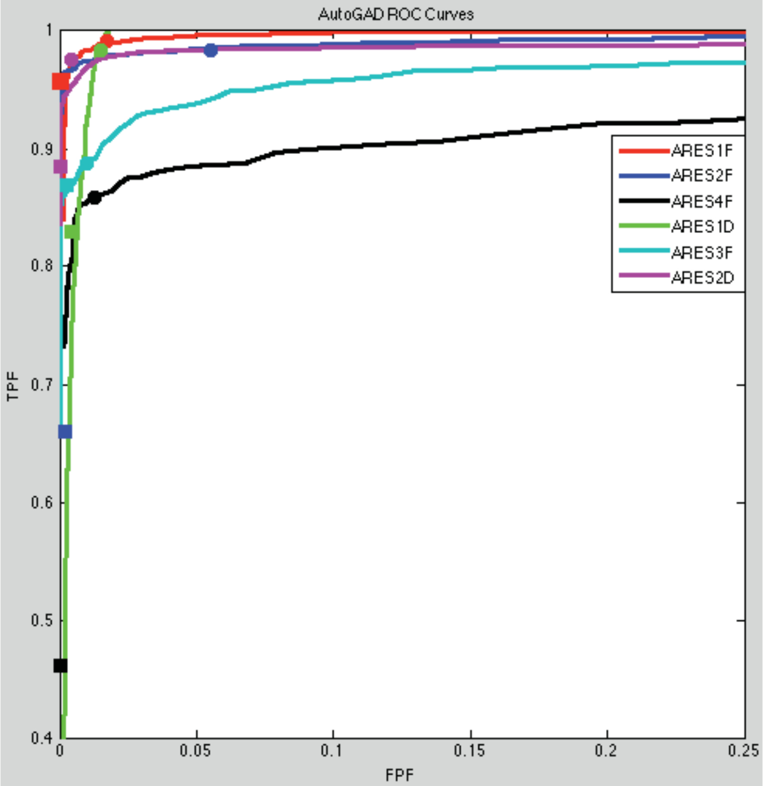


Figure 21: ROC Curves and Improved Operating Points

Considering the apparent correlation with image size eventually led to determining that the bin-width parameter needed to be automatically adjusted image by image. The original AutoGAD algorithm used a predetermined bin width, and when generating a histogram, created as many bins of this width as necessary to span from the minimum to maximum abundance for each independent component. This meant that its detection sensitivity was sensitive to both the range of the independent components and the number of pixels in the image. A way to alleviate this problem was to assign an average number of pixels per bin, Y , rather than a fixed bin width, and use this to determine the bin width dynamically for each independent component calculated,

$$\omega = \frac{Y}{n}(\min(score) - \max(score)) . \tag{30}$$

An initial setting of $Y = 300$ pixels per bin was estimated from the original bin-width suggested by Johnson and the average image size in his training set. The results of this adjustment can be seen by viewing the new operating points shown in Figure 21, shown as dots along the ROC curves. The results of this adjustment are also shown numerically, side by side, with the original algorithm results in Table 2. There is clearly a drastic improvement in TPF level and consistency. FPF also increased somewhat dramatically. To compensate for this, sensitivity can be adjusted by changing Y to affect a desired FPF limit. The effect of adjusting Y from 300 to 700 pixels per bin can readily be seen in Table 2, as FPF and TPF rates both decrease as Y increases.

Table 2: AutoGAD Improvement Results

				IMPROVED AUTOGAD					
		OLD AUTOGAD		Y=300		Y=500		Y=700	
	Pixels	Mean TPF	Mean FPF	Mean TPF	Mean FPF	Mean TPF	Mean FPF	Mean TPF	Mean FPF
ARES1F	**30560**	0.9571	0.0017	0.9906	0.0137	0.9803	0.0036	0.9728	0.0031
ARES2F	**47424**	0.6599	0.0027	0.9843	0.0637	0.9811	0.0328	0.9760	0.0202
ARES4F	**16400**	0.4610	0.0036	0.8679	0.0132	0.8520	0.0076	0.8333	0.0055
ARES1D	**57909**	0.8298	0.0034	0.9831	0.0323	0.9843	0.0240	0.9331	0.0149
ARES3F	**30736**	0.8723	0.0025	0.8742	0.0497	0.8918	0.0259	0.8768	0.0012
ARES2D	**22360**	0.8845	0.0003	0.9789	0.0043	0.9516	0.0005	0.9289	0.0004
Mean	**34232**	**0.7774**	**0.0024**	**0.9465**	**0.0295**	**0.9402**	**0.0157**	**0.9202**	**0.0076**

3.4 Logistic PCA Reconstruction Error Anomaly Detection (LogPCARD)

In this section, non-linear principal component analysis is performed via a DBN based replicator neural network to identify anomalies through reconstruction error. The replicator neural network is stochastically pre-trained and constructed using RBMs with logistic units as described in 2.3.2. This allows for gradient descent backpropagation to be efficiently performed on the resulting DBN such as to 'fine tune' the replicator neural network. Anomaly declaration is attempted before, and after backpropagation. Recall section 2.5.2, where replicator neural network reconstruction error is used to detect anomalies; the same score for anomaly detection is employed to detect hyperspectral targets in this section. Further, two other scores that work well are presented: the within pixel variance of reconstruction errors, as well as the "variance explained" for a given pixel.

3.4.1 Data Preparation

Using logistic sigmoidal units requires that the input data approximate "probabilities" when input to the first layer of visible units. For hyperspectral imagery this can be accomplished via normalization,

$$X^{N}_{n\times p} = (X_{n\times p} - \vec{1}_{n\times 1} m^{T})M^{-1}, \tag{31}$$

where

$$m = (\min_{(1)}, \min_{(2)}, ..., \min_{(p)})^T, \tag{32}$$

is a vector of the band minimum values, $\vec{1}$ is a $n \times 1$ matrix of ones, and

$$M = \begin{bmatrix} \max_{(1)} - \min_{(1)} & & & \\ & \max_{(2)} - \min_{(2)} & & \\ & & \ddots & \\ & & & \max_{(p)} - \min_{(p)} \end{bmatrix} \tag{33}$$

is a matrix with the ranges of intensities by band on its diagonal. The transformation forces all values of $X_{n\times p}^{N}$ to be bounded [0,1], and allows band intensities to be treated as probabilities and presented to the visible units in the binary RBM energy function. In addition to this transformation, absorption bands are removed as described earlier in 2.2.

3.4.2 DBN Formation and Training Parameters

As in any neural network, the structure of the deep belief network is of key importance. There must be sufficient structure to represent the data, but also not superfluous units such as to dilute signal. Furthermore, redundant logistic principal components lose value for interpretation. An autoencoder structure found by trial and error to be useful is shown on the right of Figure 22, and will be used in the examples throughout this section. Recall from section 2.3.2.4, that the autoencoder is constructed from the pre-trained RBMs on the left. The architecture in Figure 22 seemed to balance representational power and speed, although anomaly detection through Log PCA reconstructions seemed to be very robust to network structure. The number of hidden units in the first through fourth (and last) sequentially trained RBMs are as follows: $A = 125$, $B = 75$, $C = 35$, and the top layer of logistical principal components $D = 3$.

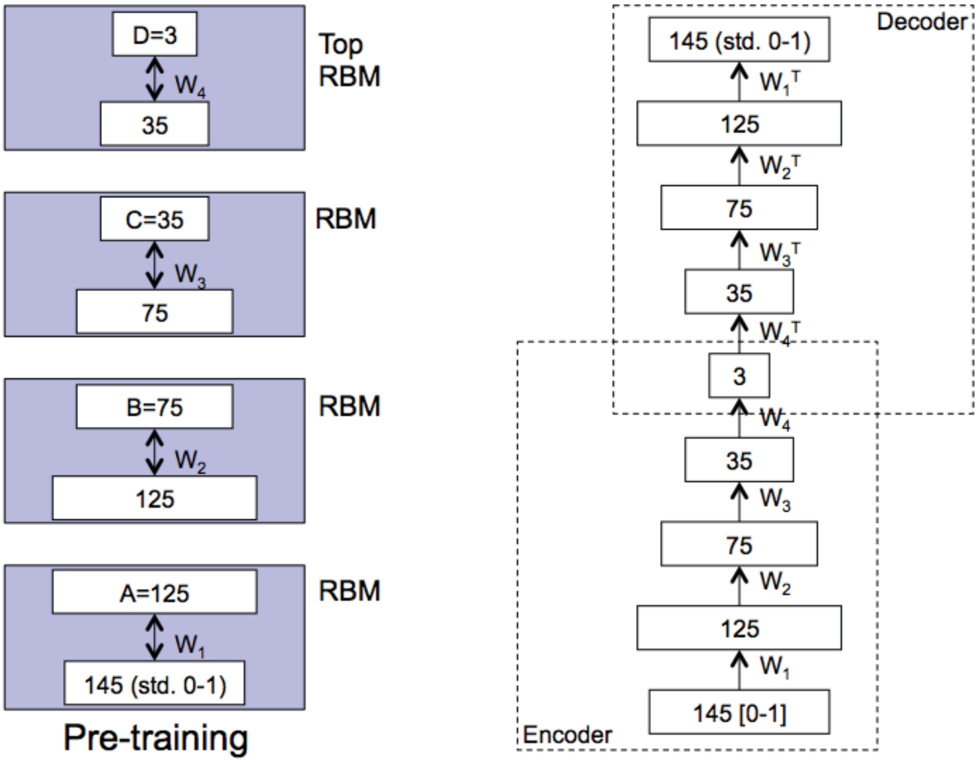


Figure 22: HSI Log PCA Recon Anomaly Detector Structure

The algorithm used to train the RBMs employs contrastive divergence and mirrors the code presented by (Hinton G. , 2007). Parameters for training are listed in Table 3. The parameters ε_w , ε_a , and ε_b are the learning rates as described in (10) and (11), whereas w_c , $p_{initial}$, and p_{final} are the weight-cost, initial momentum, and final momentum, respectively. Weight-cost is a penalty for high connection weights generated during training and reduces the learning rate to avoid over-fitting and 'sticking' on or off of the binary units, while reducing possible errors introduced by under sampling of the Markov chain (Hinton G. E., 2010). This is important because using contrastive divergence only accounts for, at most, the first few steps in the Markov chain. The momentum terms are used to change the step size during training as the gradient becomes smaller towards the end of training and progress slows. The resultant update, ΔW, during each step, k, is

$$\Delta W_k = p \cdot \Delta W_{k-1} + \varepsilon_w \left(\Delta W - W_{k-1} \cdot w_c \right) \tag{34}$$

where ΔW is the matrix of weight updates calculated via (10).

The center column in Table 3 holds training parameters for the first RBM A, while RBM B, C, and D's parameters are shown in the right column. A maximum of 30 training epochs was chosen. To enable a stopping rule, the total training error was monitored at each epoch. A linear regression was performed on the last three epoch's total reconstruction errors, and training was stopped when the slope became greater than or equal to zero. Using reconstruction error is not the optimal method for monitoring training, (Hinton, 2010). but it is useful in that it is easy to compute and use for automation in the anomaly detection application. In order to speed up training, updates are calculated in batches rather than individually (Hinton, 2010). A batch size of between 10-100 was computed depending on the size of the image being studied. Larger images had larger batch sizes such as to increase efficiency and to avoid overtraining.

Table 3: RBM Training Settings

Parameter	D	A, B, & C
ε_w	.003	.2
ε_a	.003	.2
ε_b	.003	.2
w_c	$9e-5$	.003
$p_{initial}$	.6	.65
p_{final}	.9	.85
max_{Epoch}	30	30
$batchsize$	$10-100$	$10-100$

Recall from 2.3.2 that RBM A, B, C, and then D will be trained sequentially; using the outputs from the hidden layers in A as inputs to the visible layers in B and so on. The resultant structure is then ‘stacked’ and ‘unfolded’ into an encoder and decoder for the logistic principal components as shown in Figure 22. At this point, reconstructions of the original data points, $x_{1\times p} \in X^{S}_{n\times p}$ may be obtained by first encoding the logistic principal components, $T_{\log}$, as shown in Figure 23. Here, B_{1-4} are matrices of the hidden biases, and z_{1-4} are vectors of the binary output probabilities at each layer.

Layer 1

$$z_{1(1\times k_1)} = \frac{1}{1+\exp\left(-\begin{bmatrix} x_{1\times p} & 1 \end{bmatrix}\begin{bmatrix} W_{1(p\times k_1)} \\ B_{1(1\times k_1)} \end{bmatrix}\right)}$$

Layer 2

$$z_{2(1\times k_2)} = \frac{1}{1+\exp\left(-\begin{bmatrix} z_{1(1\times k_1)} & 1 \end{bmatrix}\begin{bmatrix} W_{2(k_1\times k_2)} \\ B_{2(1\times k_2)} \end{bmatrix}\right)}$$

Layer 3

$$z_{3(1\times k_3)} = \frac{1}{1+\exp\left(-\begin{bmatrix} z_{2(1\times k_2)} & 1 \end{bmatrix}\begin{bmatrix} W_{3(k_2\times k_3)} \\ B_{3(1\times k_3)} \end{bmatrix}\right)}$$

Layer 4

$$z_{4(1\times k_4)} = \begin{bmatrix} z_{3(1\times k_3)} & 1 \end{bmatrix}\begin{bmatrix} W_{4(k_3\times k_4)} \\ B_{4(1\times k_4)} \end{bmatrix}$$

$$T_{n\times k_4} = z_4 \forall x \in X^{S}_{n\times p}$$

Figure 23:Log PCA Encoding

A prediction is then obtained by decoding the Log PCs as shown in Figure 24, where A_{1-4} are matrices of the visible biases, z_{5-7} are again vectors of the binary output probabilities, and $\hat{X}^{W}_{n\times p}$ is a matrix of the reconstructed pixels, $\hat{x}$.

Layer 5

$$z_5 = \frac{1}{1+\exp\left(-\begin{bmatrix} z_{4(1\times k_4)} & 1 \end{bmatrix}\begin{bmatrix} W_4^T \\ A_{4(1\times k_3)} \end{bmatrix}\right)}$$

Layer 6

$$z_6 = \frac{1}{1+\exp\left(-\begin{bmatrix} z_{5(1\times k_3)} & 1 \end{bmatrix}\begin{bmatrix} W_3^T \\ A_{3(1\times k_2)} \end{bmatrix}\right)}$$

Layer 7

$$z_7 = \frac{1}{1+\exp\left(-\begin{bmatrix} z_{6(1\times k_2)} & 1 \end{bmatrix}\begin{bmatrix} W_2^T \\ A_{4(1\times k_1)} \end{bmatrix}\right)}$$

Layer 8

$$\hat{x} = \frac{1}{1+\exp\left(-\begin{bmatrix} z_{4(1\times k_1)} & 1 \end{bmatrix}\begin{bmatrix} W_1^T \\ A_{4(1\times p)} \end{bmatrix}\right)}$$

$$\hat{X}_{n\times p}^{W} = \left[\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n\right]^T$$

Figure 24: Log PCA Decoding

Finally, thirty epochs of conjugate gradient backpropagation are performed on the full encoder-decoder structure to ‘fine-tune’ the weights. Reconstructions can then be obtained in the same manner as before using the ‘fine-tuned’ autoencoder. Comparisons of the reconstruction errors for use in anomaly detection before and after backpropagation are presented in the following section.

3.4.3 Log PCA Reconstruction Anomaly Detection on ARES1D and ARES1F

Three scores derived from logistic principal component reconstructions are compared in this section. All require the calculation of the reconstruction errors (“residuals”), R , for each pixel,

$$R_{i(1\times p)} = X_i^N - \hat{X}_i^N \quad \forall i = 1,2,\ldots,n \qquad (35)$$

The first score is the essentially the same as the δ_i score outlined in section 2.3.4, and the Q_{PCA} score to be outlined in section 3.3. In the case of Log PCA, the sum of squared reconstruction errors, $Q_{\log PCA}$ may be obtained by

$$Q_{i(\log PCA)} = R_{i(1\times p)} R^T_{i(1\times p)} = \left(X_i^N - \hat{X}_i^N\right)\left(X_i^N - \hat{X}_i^N\right)^T \quad \forall i = 1,2,...,n. \tag{36}$$

The second score is the variance, $\sigma^2_{R_i}$, of the "residuals" across bands within a pixel and is calculated

$$\sigma^2_{R_i} = \frac{1}{p}\sum_{i=1}^{p}\left(R_i - \mu_{R_i}\right)^2 \quad \forall i = 1,2,...,n. \tag{37}$$

where, μ_{R_i} ,is the mean reconstruction error within a pixel (again calculated across bands).

The final score, R^{2*}, is akin to the r-squared term in a linear regression. The score is the variance of the residuals, $\sigma^2_{R_i}$, divided by the "variance" across band radiances, r, within a pixel,

$$\sigma^2_{r(pixel)} = \frac{1}{p}\sum_{i=1}^{p}\left(r_i - \mu_{r(pixel)}\right)^2 \tag{38}$$

A vector of the variance explained score, R, is then calculated

$$R_i^{2*} = \frac{\sigma^2_{R_i}}{\sigma^2_{r_i}} \quad \forall i = 1,2,...,n. \tag{39}$$

It should be noted that the R_i^{2*} value differs from that of an r-squared term in a linear regression as the $\sigma^2_{R_i}$ may exceed the $\sigma^2_{r_i}$. In this way it is not calculated as $1 - R_i^{2*}$ as this value is not logical, and for anomaly detection purposes it might actually be better to consider the score "variance not explained."

Figure 25 and Figure 26 on the following pages show the utility of the above three scores for anomaly detection before and after 'fine-tuning' on ARES1D. The top row of plots in each figure show results for the Q score, while the middle shows $\sigma^2_{R_i}$, and the bottom row R_i^{2*}. The scores have been normalized in the scatterplot on the left and a horizontal line at 3 standard deviations is drawn for reference. It is interesting to note the decreased noise apparent in the abundance plot, and the improvements in ROC moving from Q to $\sigma_{R_i}^2$, to R_i^{2*} in Figure 25 showing results before backpropagation. The R^{2*} statistic plots show a drastic reduction in the effect of the sagebrush and noise on anomaly detection, but the abundance plot seems to show some shape distortions of the original anomalies.

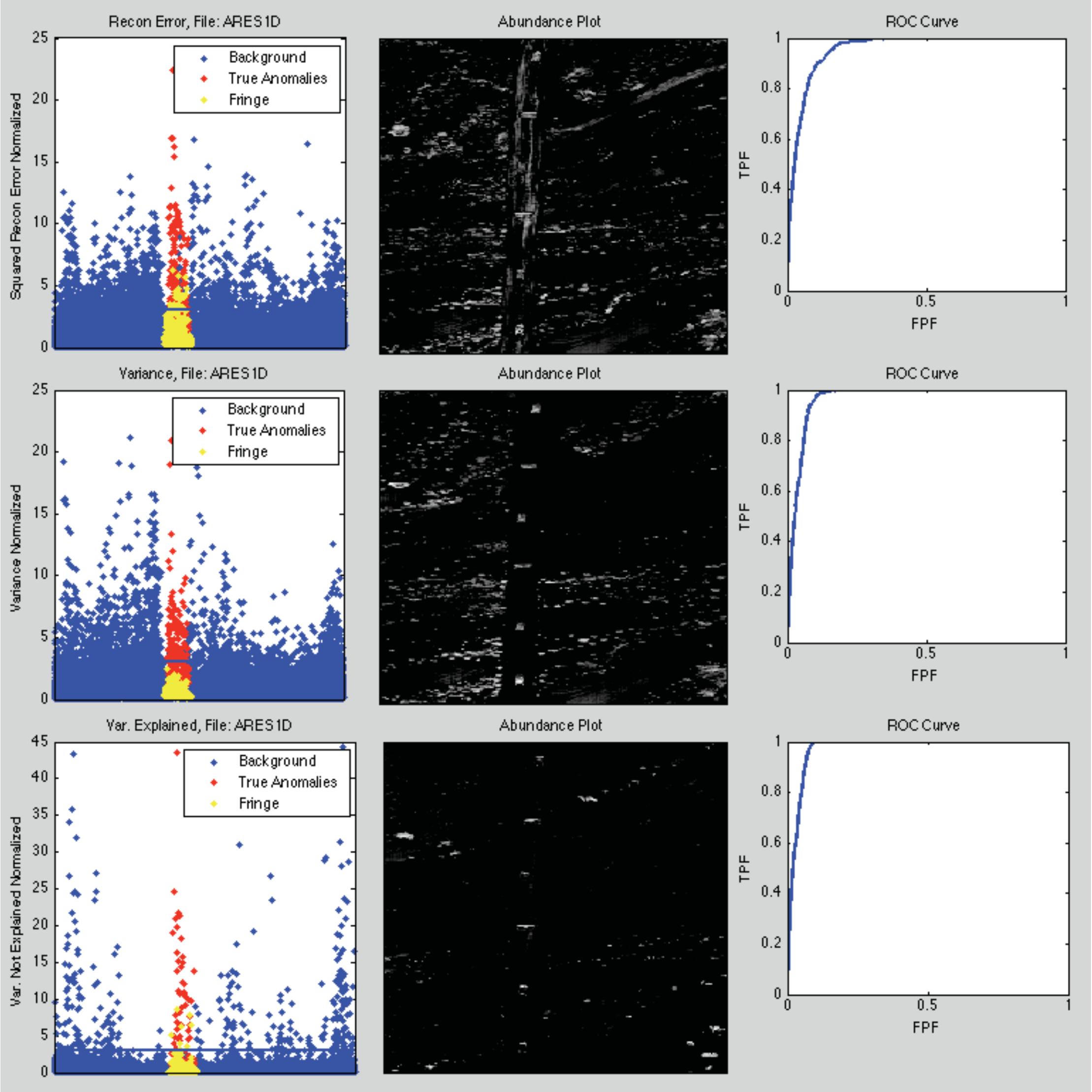


Figure 25: Log PCA Recon. Anomaly Detection ARES1D, Before 'fine-tuning'

Interestingly, backpropagation seemed to negatively affect anomaly detection with the R^{2*} statistic in Figure 26, while performance improved for σ_R^2 and Q. A possible explanation may lie in the R^{2*} abundance plot. Here, the intensity of the top five vehicles seems to have increased, while the bottom vehicle disappeared almost completely. It seems probable that over-training caused this problem, although the bottom vehicle appears to have increased in intensity in the σ_R^2 and Q abundance plots.

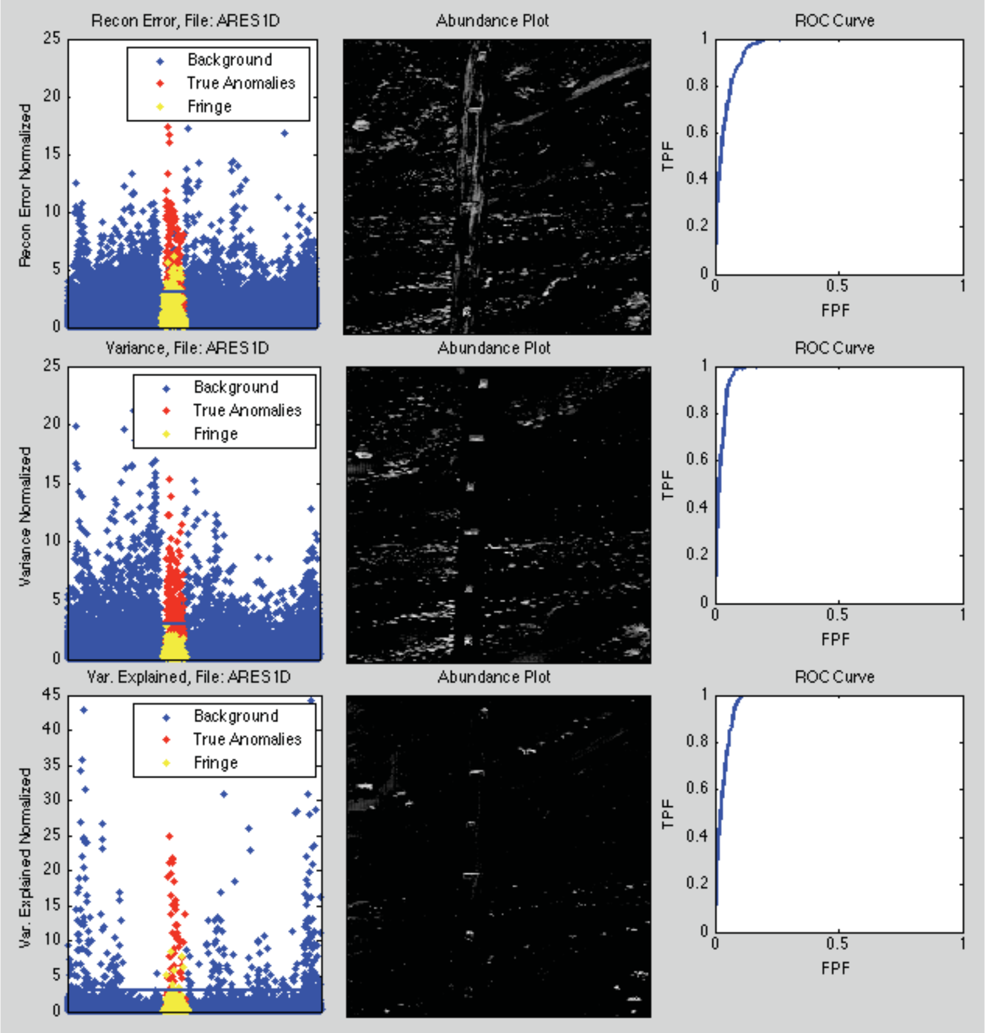


Figure 26: Log PCA Recon. Anomaly Detection ARES1D, After 'fine-tuning'

Figure 27 shows similar results after before backpropagation for ARES1F. Once again, ROCs improve dramatically in order from Q to σ_R^2 to R^{2*} as background clutter abundance is reduced in relation to the noise. Interestingly, in the R^{2*} abundance plot, the row of trees in the bottom left corner dominate all the anomalies as pixels so much so that the rest of the anomalies are not visible. The ROC are robust to this, but it seems that finding a consistent operating point may prove difficult with this score.

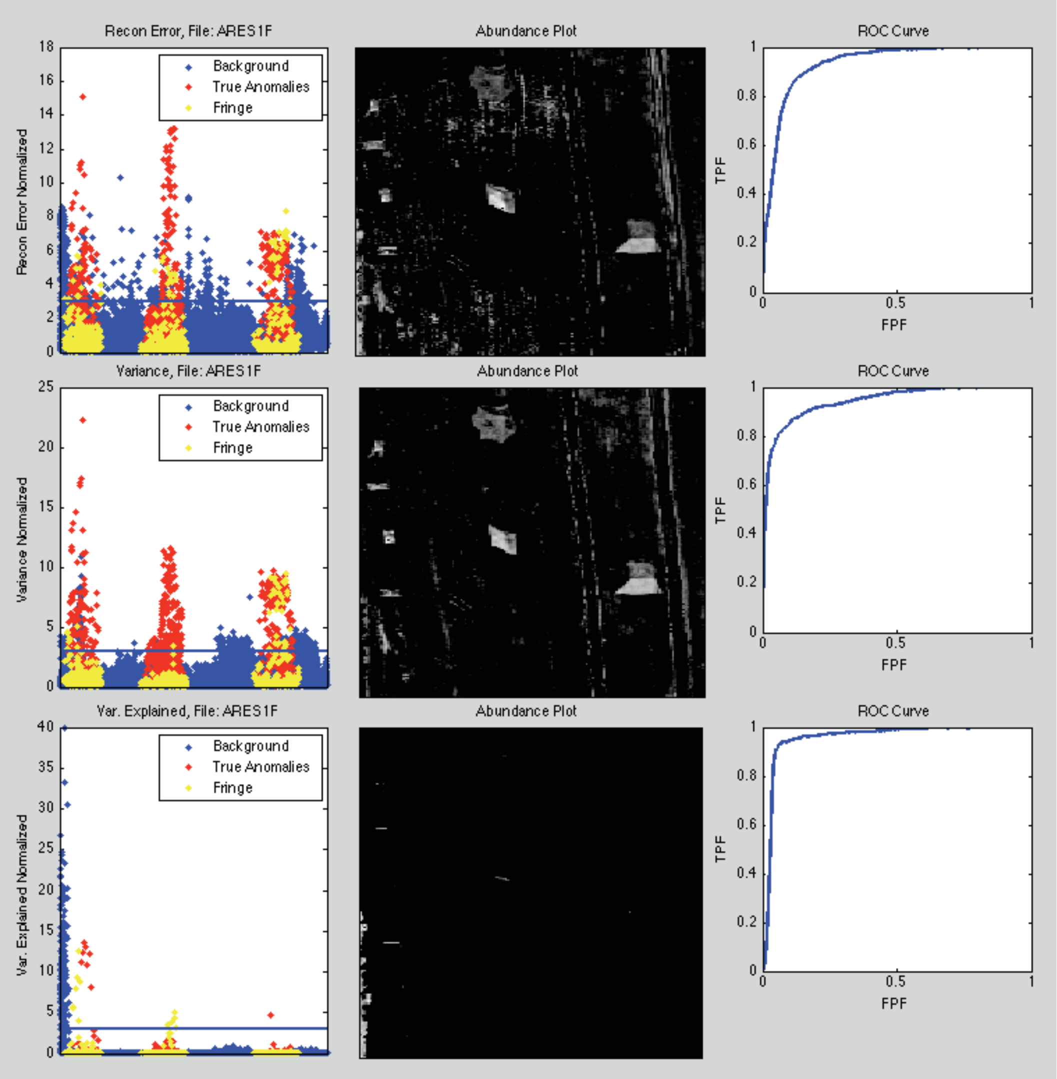


Figure 27: Log PCA Recon. Anomaly Detection ARES1F, Before ‘fine-tuning’

Finally, in Figure 28 there is a drastic improvement in ROC after backpropagation for all statistics. The abundance plots all look cleaner and it seems that the autoencoder has ‘learned’ the background pixels very well in comparison to the anomalous pixels. The problem with the trees in the bottom left corner for R^{2*} is not really fixed, but each score’s potential for use in anomaly detection were improved.

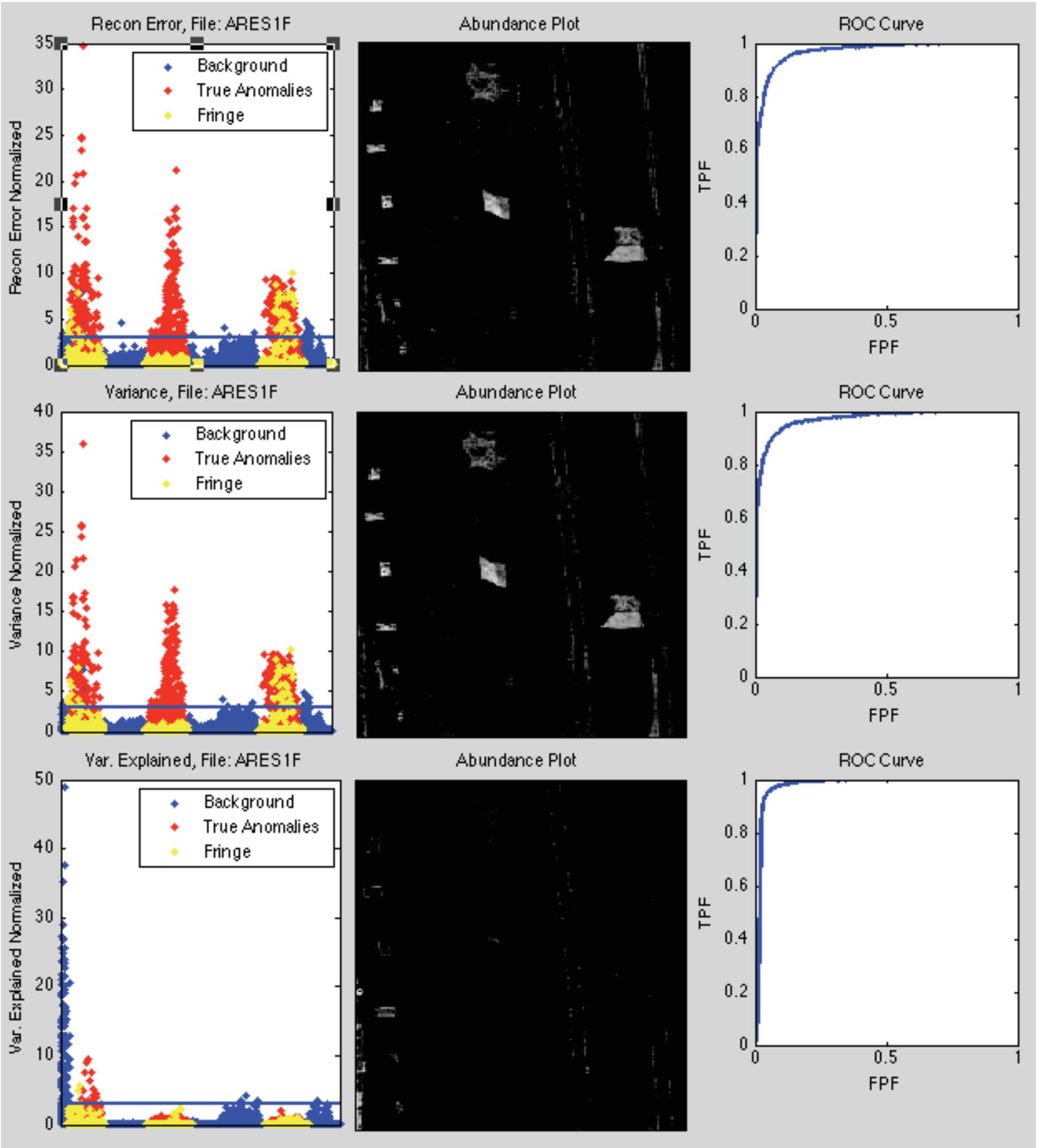


Figure 28: Log PCA Recon. Anomaly Detection ARES1F, After 'fine-tuning'

3.4.4 Zero-detection Histogram Thresholds

As was done with the AutoGAD improvements in section 3.3, an image size adaptive bin size parameter, Y, is selected and anomaly detection is performed using the zero-detection histogram method (19) outlined in section 2.4.3,

$$\left(Score_i > \vartheta_{\text{Location of First Empty Histogram Bin}}\right) \rightarrow X_i \in anomalies\ . \qquad (40)$$

The effects of different Y on the anomaly detection for the σ_R^2 score (before backpropagation) are shown in Figure 29. It is quite obvious that bin size, ω, will drastically affect the TPF and FPF rates.

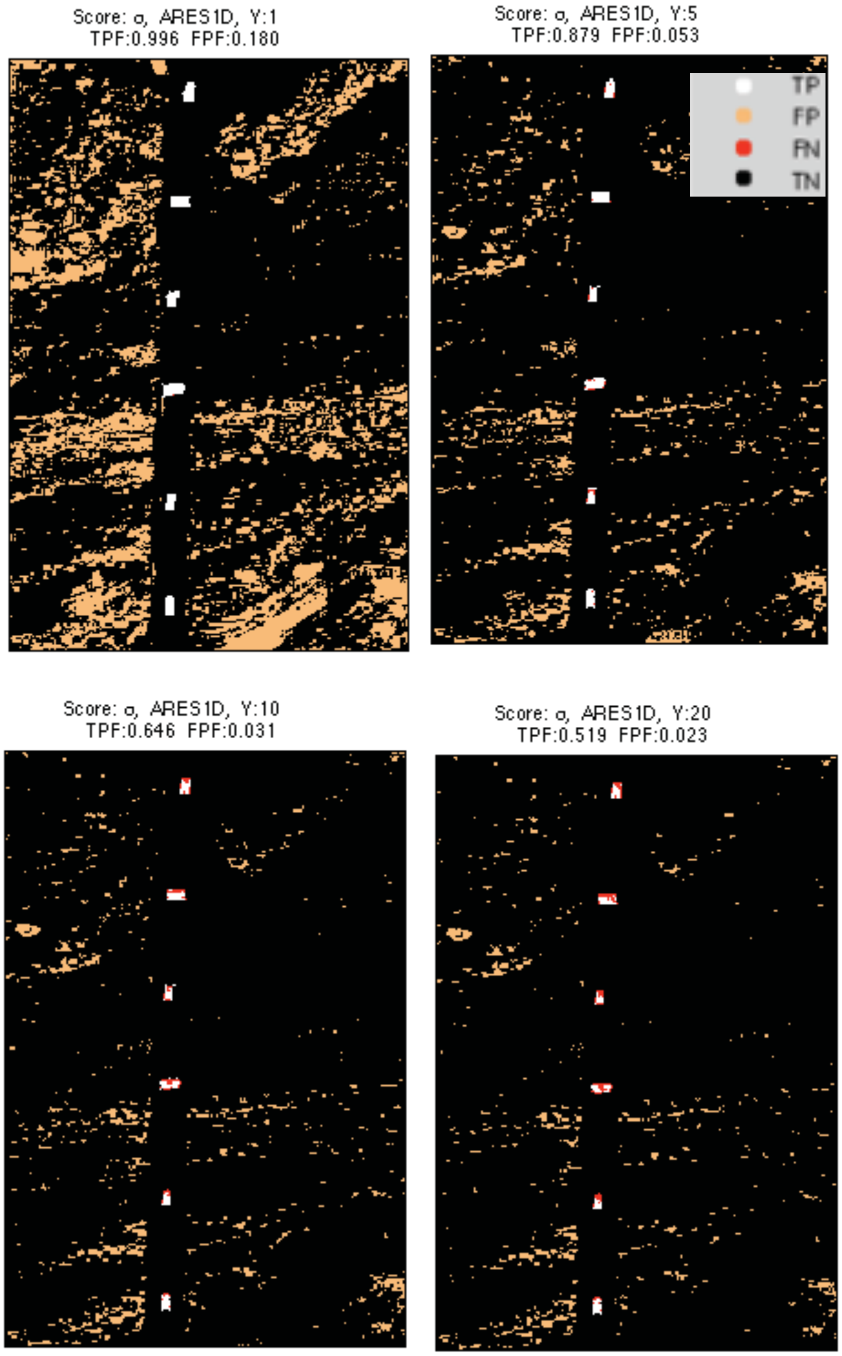


Figure 29: Varying Bin Size for ARES1D

Anomaly declarations using the zero-bin detection method for all three proposed scores using bin size parameter Y =10 are presented in Figure 30 through Figure 33. The results for ARES1F Forest 1 show the inability of the zero-bin detection method to separate the anomalies in a situation such as in the R^{2*} score where there were a number of far outlying values (the trees in the bottom left corner) as displayed in Figure 27 and Figure 28. This effectively 'squeezed' a majority of the pixels, to include the actual anomalies into the far left of the histogram and forced the detection threshold too far right. Otherwise, results for ARES1F seem modest but it appears that an appropriately selected Y could yield promising results. The results for ARES1D in Figure 32 and Figure 33 shows a more consistency across the different detection scores, although it seems that different Y values are needed for different scores. It is apparent in both images that backpropagation reduced FPF rates as the autoencoder better 'learned' the background structure of the data.

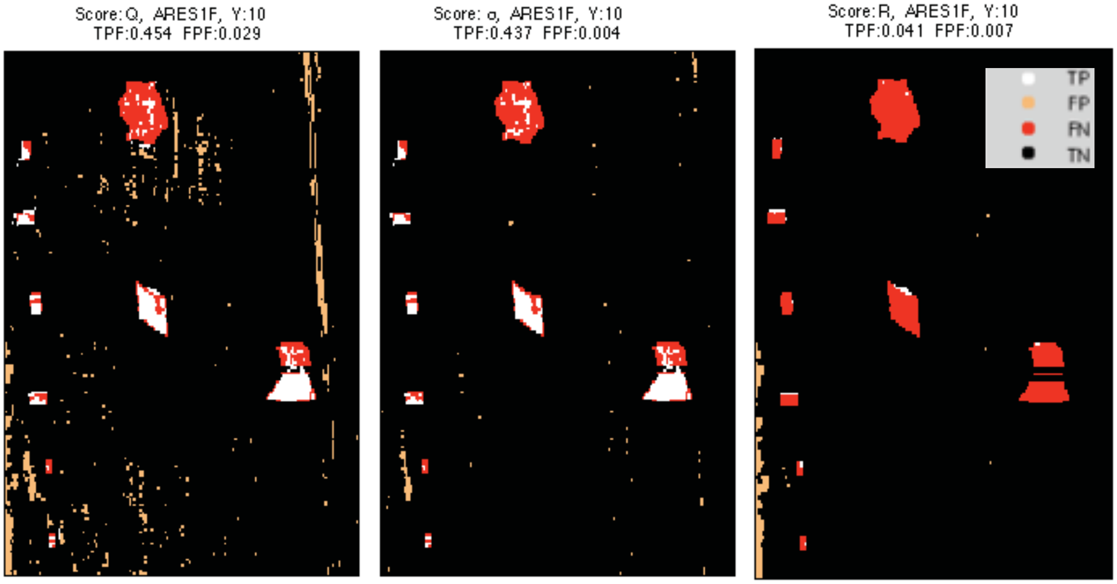


Figure 30: ARES1F LogPCARD Results Before Backpropagation

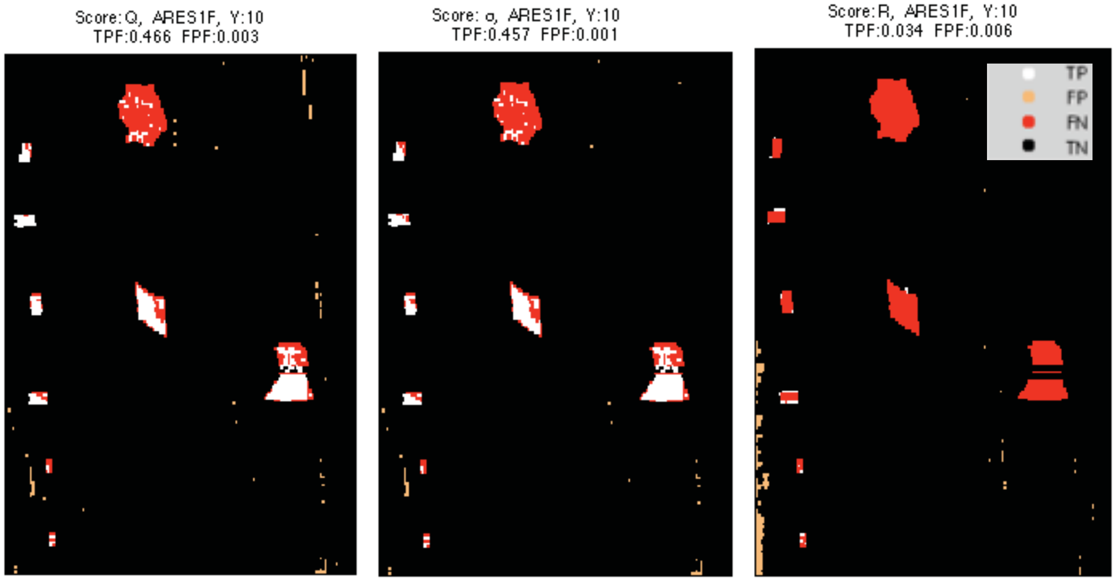


Figure 31:ARES1F LogPCARD Results After Backpropagation

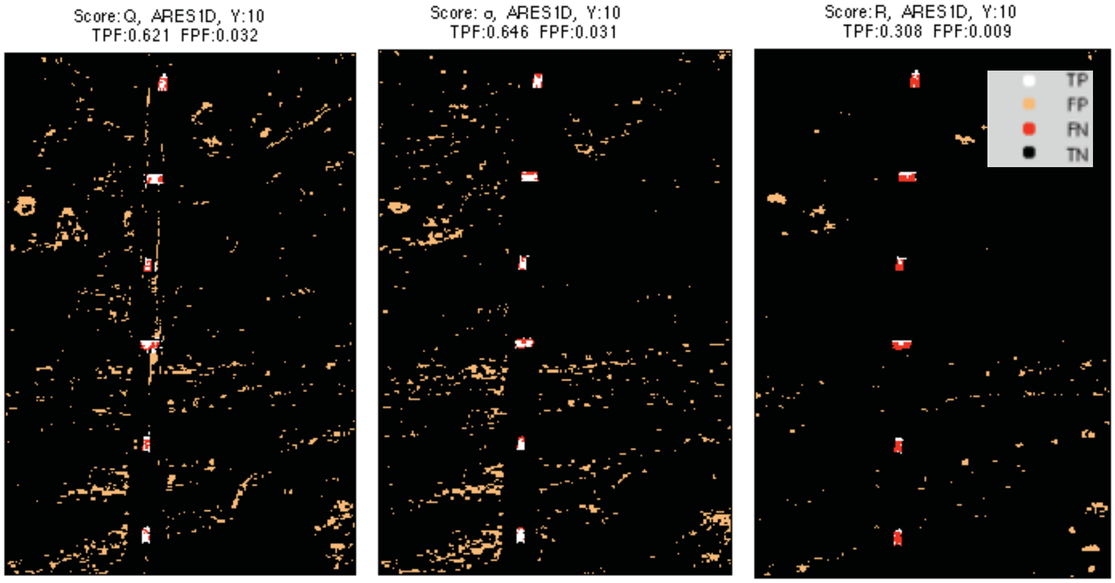


Figure 32: ARES1D LogPCARD Results Before Backpropagation

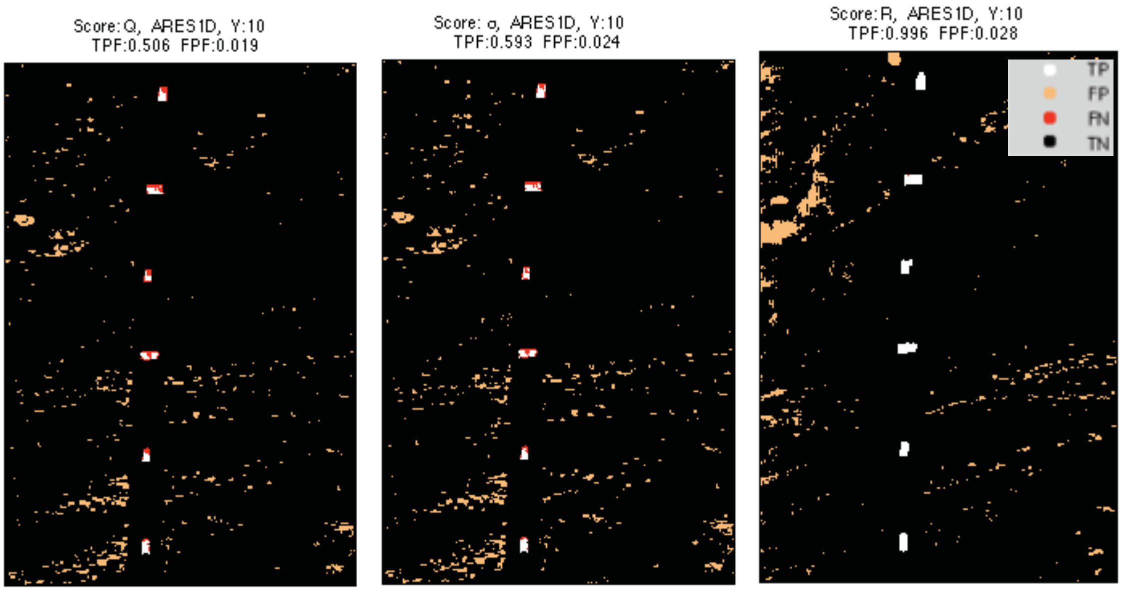


Figure 33: ARES1D LogPCARD Results After Backpropagation

3.4.5 Summary

This section presented a method for constructing and training a DBN autoencoder that successfully generated reconstructions used for HSI anomaly detection. Three different anomaly scores generated from reconstruction errors show potential for anomaly declaration with the R^{2*} score showing generally the best ROC, but inconsistency in automation due to its high variability. For this reason, the σ_R^2 score will be considered when comparing the algorithm performance with the other techniques in Chapter 4.

3.5 Global Iterative PCA Reconstruction Error Based HSI Anomaly Detection

The global iterative PCA reconstruction error based anomaly detection (GIPREBAD) method introduced in this section utilizes the squared reconstruction error statistic, Q, outlined in 2.5.1 in an iterative fashion. The iterative feature reduces extreme anomaly effects on first and second order statistic estimation, thus resulting in a more accurate estimation of the background covariance structure. This is a similar concept to that employed by Taitano, Geier, and Bauer (2010) for constructing the locally adaptable iterative RX detector.   Subsequent iterations of GIPREBAD increase the total reconstruction error for anomalies and thus separate targets from the background for detection.

The results of iteratively removing suspected anomalies from the covariance estimation are readily apparent in Figure 34 showing the 1st, 2nd, 4th, and 10th iteration of GIPREBAD on ARES1F. Referencing the truth mask in Figure 19, notice the increasingly defined appearance of known anomalies on the abundance plot after each iteration. Further, the ROC curves on the right show the ability of the algorithm to affect accurate classification of anomalies.

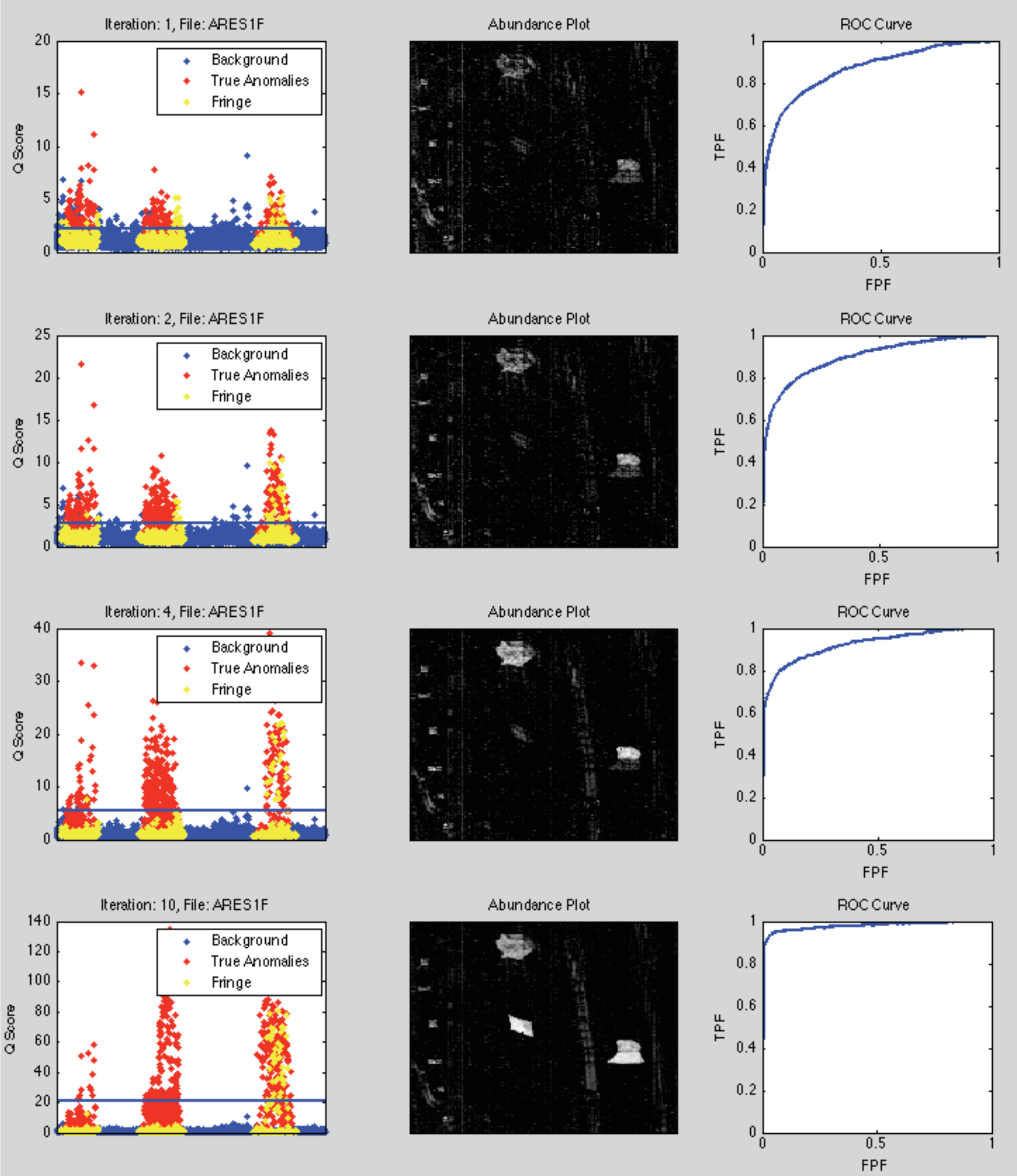


Figure 34: GIPREBAD Iterations 1, 2, 4, & 10, ARES1F

GIPREBAD starts by standardizing and centering the reshaped hyperspectral data, $X_{n \times p}$ prior to determining the principal component directions and the magnitude of their associated eigenvalues. The transformation ensures equal weighting between spectral bands that might otherwise maintain unequal scaling and is equivalent to using the sample correlation matrix in place of the covariance matrix in the PCA formulation. This affine transformation is performed as

$$X^S_{n\times p} = (X_{n\times p} - \vec{1}_{1\times p}\mu^T)D^{-1/2} \tag{41}$$

where

$$\mu = (\mu_{(1)}, \mu_{(2)}, \ldots, \mu_{(p)})^T \tag{42}$$

is a vector of the band means, u is a $n\times 1$ matrix of ones, and

$$D = \begin{bmatrix} \sigma^2_{(1)} & & & \\ & \sigma^2_{(2)} & & \\ & & \ddots & \\ & & & \sigma^2_{(p)} \end{bmatrix} \tag{43}$$

is a diagonalized matrix of the band variances. The result is the matrix $X^S_{n\times p}$, of hyperspectral pixels, standardized and centered by band. A dimensionality, k, assessment is then made according to Kaiser's criterion as PCA is performed. The resultant k principal components, T, are then computed where

$$T_{n\times k} = X^S_{n\times p}V_{p\times k}. \tag{44}$$

The principal components are then used to project PCs back into the original data structure as a reconstruction, $X^S_{n\times p}$, where

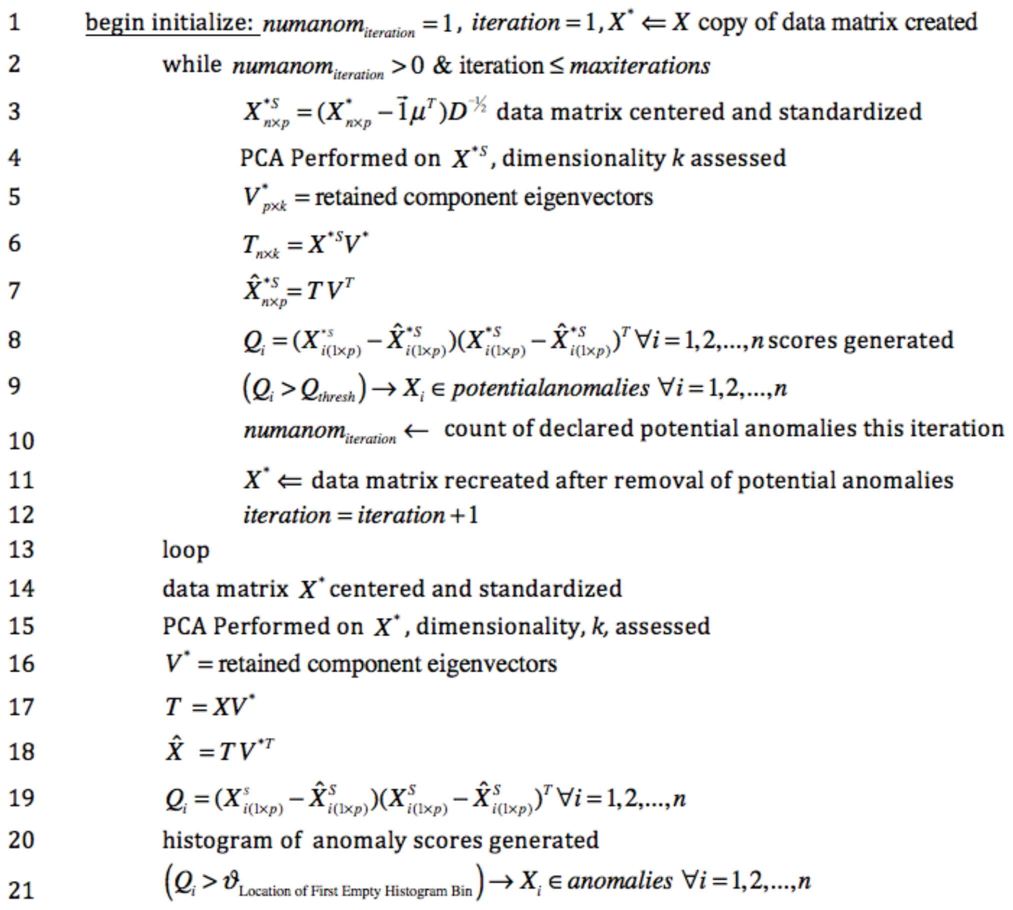
$$\hat{X}^S_{n\times p} = T_{n\times k}V^T_{p\times k} \tag{45}$$

This reconstruction is then subtracted from the original data vector to form a vector of residuals. These residuals are squared and summed, forming the approximately normally distributed score Q as in (17)

$$Q_i = (X^s_{i(1\times p)} - \hat{X}^S_{i(1\times p)})(X^S_{i(1\times p)} - \hat{X}^S_{i(1\times p)})^T \,\forall i = 1, 2, \ldots, n\,. \tag{46}$$

Consider the mean of this statistic to be μ_Q and the standard deviation σ_Q. Exemplars where $Q > Q_{thresh}$, where $Q_{thresh} = \mu_Q + 2.801 \cdot \sigma_Q$ are added to the set O, of potential outliers. The value chosen for Q_{thresh} is essentially a nominal declaration threshold loosely justified by its location at the .995 normal quantile (assuming that the outliers represent a small percentage of the data).

A new subset of the standardized data, $\hat{X}^S_{n \times p}$, is now formed that does not include the potential outliers declared in O. This new data set will be used to estimate the background covariance structure and again form principal components; beginning a new iteration where potential anomalies are once again excluded. The algorithm reassesses dimensionality and generates a new Q_{thresh} during each iteration. Iterations are continued until no new anomalies are declared or a pre-defined maximum limit of iterations is reached. Pseudo code outlining the iterative procedure is shown in Figure 35.



```
begin initialize: $numanom_{iteration} = 1$, $iteration = 1, X^{*} \Leftarrow X$ copy of data matrix created
    while $numanom_{iteration} > 0$ & iteration $\leq maxiterations$
        $X^{*S}_{n\times p} = (X^{*}_{n\times p} - \vec{1}\mu^{T})D^{-1/2}$ data matrix centered and standardized
        PCA Performed on $X^{*S}$, dimensionality $k$ assessed
        $V^{*}_{p\times k}$ = retained component eigenvectors
        $T_{n\times k} = X^{*S}V^{*}$
        $\hat{X}^{*S}_{n\times p} = TV^{T}$
        $Q_i = (X^{*s}_{i(1\times p)} - \hat{X}^{*S}_{i(1\times p)})(X^{*S}_{i(1\times p)} - \hat{X}^{*S}_{i(1\times p)})^{T}\ \forall i = 1,2,\ldots,n$ scores generated
        $(Q_i > Q_{thresh}) \rightarrow X_i \in potentialanomalies\ \forall i = 1,2,\ldots,n$
        $numanom_{iteration} \leftarrow$ count of declared potential anomalies this iteration
        $X^{*} \Leftarrow$ data matrix recreated after removal of potential anomalies
        $iteration = iteration + 1$
    loop
    data matrix $X^{*}$ centered and standardized
    PCA Performed on $X^{*}$, dimensionality, $k$, assessed
    $V^{*}$ = retained component eigenvectors
    $T = XV^{*}$
    $\hat{X} = TV^{*T}$
    $Q_i = (X^{s}_{i(1\times p)} - \hat{X}^{S}_{i(1\times p)})(X^{S}_{i(1\times p)} - \hat{X}^{S}_{i(1\times p)})^{T}\ \forall i = 1,2,\ldots,n$
    histogram of anomaly scores generated
    $(Q_i > \vartheta_{\text{Location of First Empty Histogram Bin}}) \rightarrow X_i \in anomalies\ \forall i = 1,2,\ldots,n$
```

Figure 35:Pseudocode for GIPREBAD

After the final anomaly scores, Q, for all pixels within the image are calculated. The zero-detection histogram method outlined in section 2.4.3 is then used to pick an appropriate threshold for anomaly declarations. Zero-detection is not used during individual iterations as it slows the algorithm due to the high computational demands of repeated sorting, while also not yielding significant gains in accuracy. Final ROC curves for the GIPREBAD algorithm on ARES1D and ARES1F are shown in Figure 36 and Figure 37.

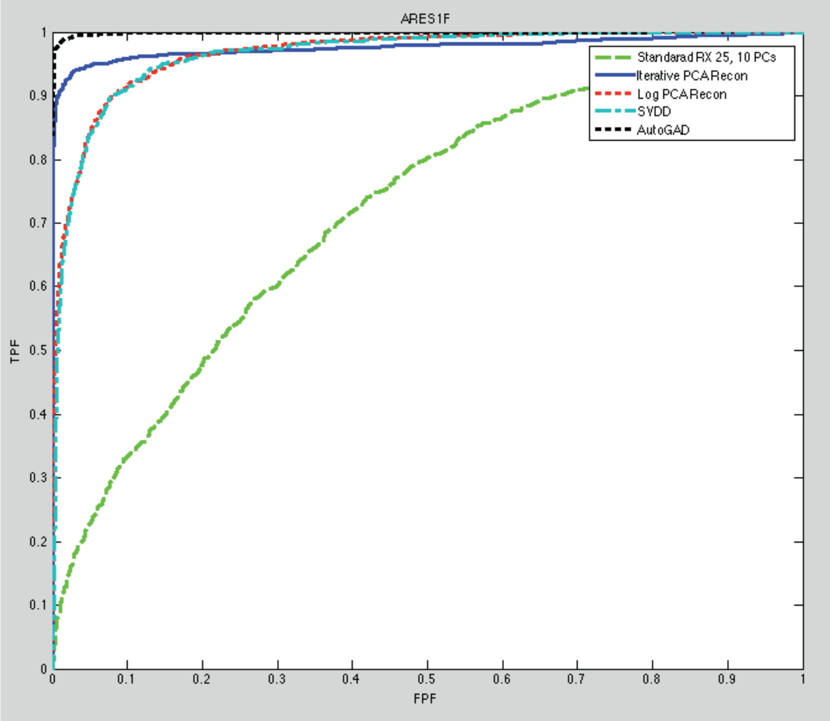


Figure 36: GIPREBAD Relative Performance, ARES1F

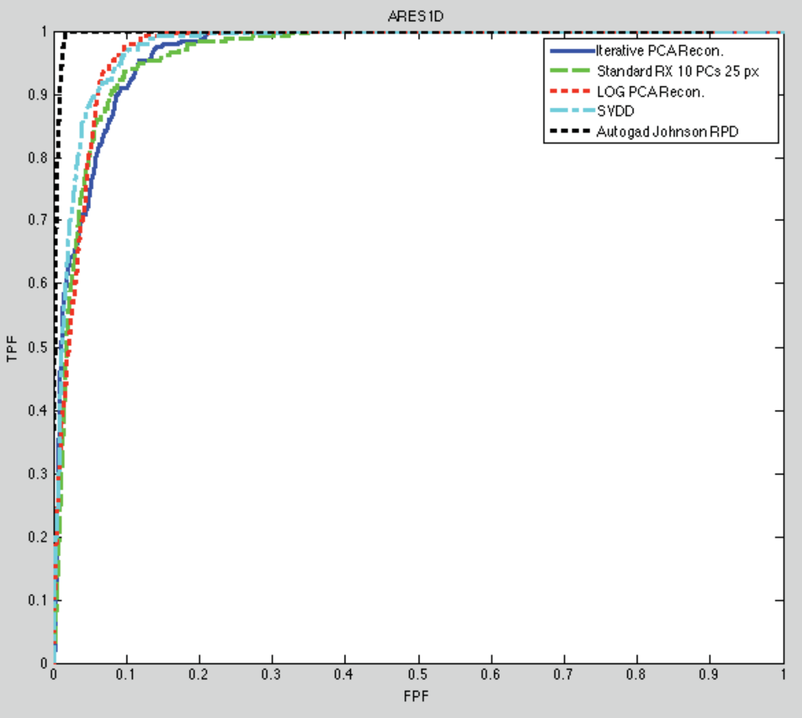


Figure 37: GIPREBAD Relative Performance, ARES1D

GIPREBAD performed favorably on ARES1F providing excellent ROC when compared to the benchmark standard RX algorithm, SVDD, and Log PCA Recon only to be outperformed by AutoGAD as shown in Figure 36. On the other hand, nearly all other algorithms presented for comparison outperform GIPREBAD on ARES1D (although it is still quite competitive). Figure 38 showing the 1st, 2nd, 3rd, and 10th GIPREBAD iterations for ARES1D offer some insight into this outcome. Note in the first iteration Q_{score} plot how an abundance of background pixels are more poorly reconstructed than the known anomalies. One possible explanation for this is that the targets may have been reconstructed well compared to pixels in the background due to a large amount of anomalous spectral information being included in the retained principal components (too many components retained). Alternatively, the background pixels that are being poorly reconstructed may not be represented well in the components retained (too few components retained).

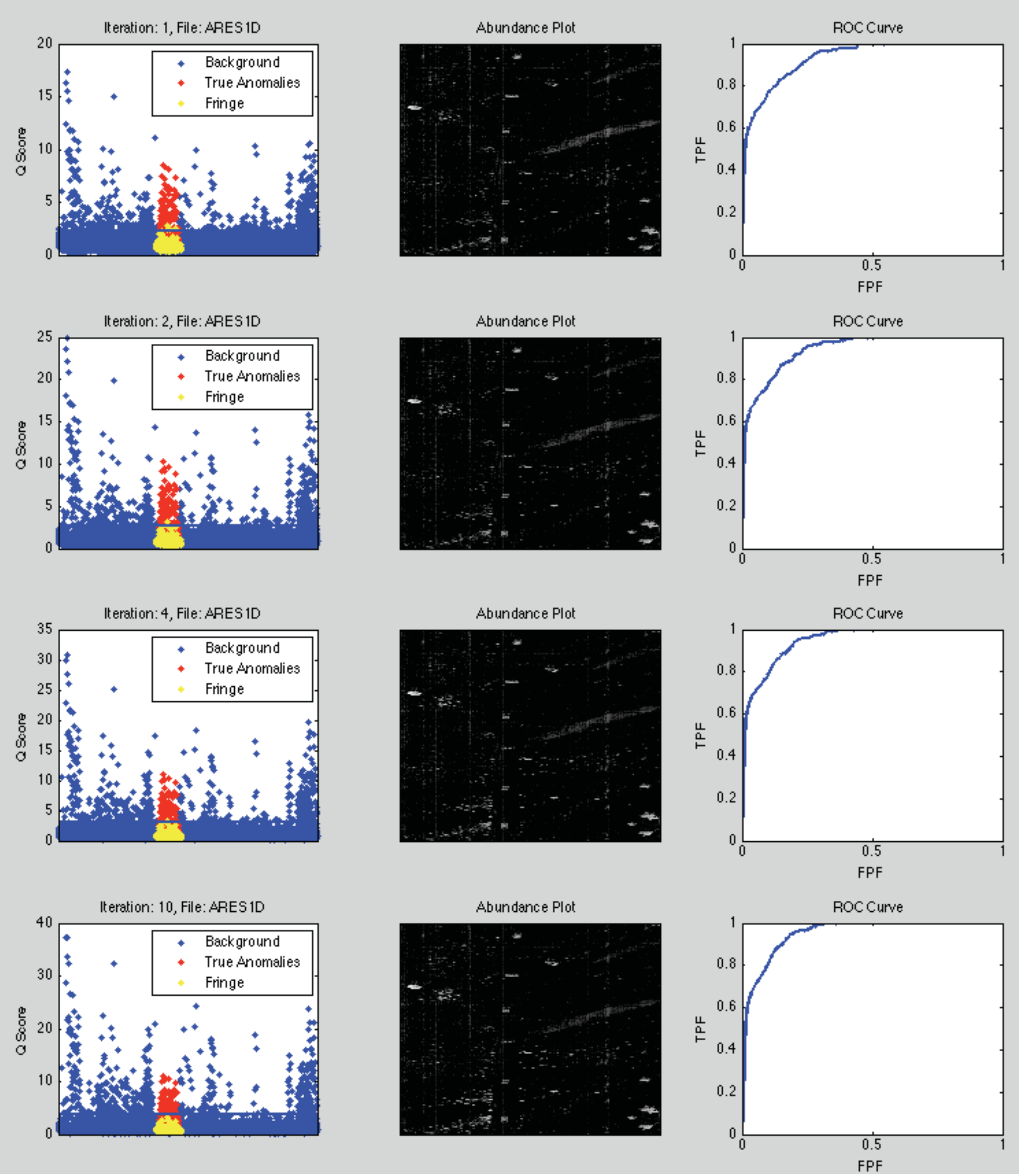


Figure 38: GIPREBAD, 1, 2, 4, & 10 Iterations, ARES1D

In order to adjust for a potential systemic over or under estimation of dimensionality causing performance problems, a correction factor, c_k, is created. After obtaining dimensionality assessment through Kaiser's criterion, the result is adjusted as $k = k + c_k$. ROC curves for nine different levels of c_k tested in the global iterative PCA reconstruction method for ARES1D and ARES1F are shown in Figure 39. There is little evidence that a correction to the dimensionality assessment will enable better anomaly declaration for images similar to ARES1D; therefore, perhaps noise is the primary cause of underperformance.

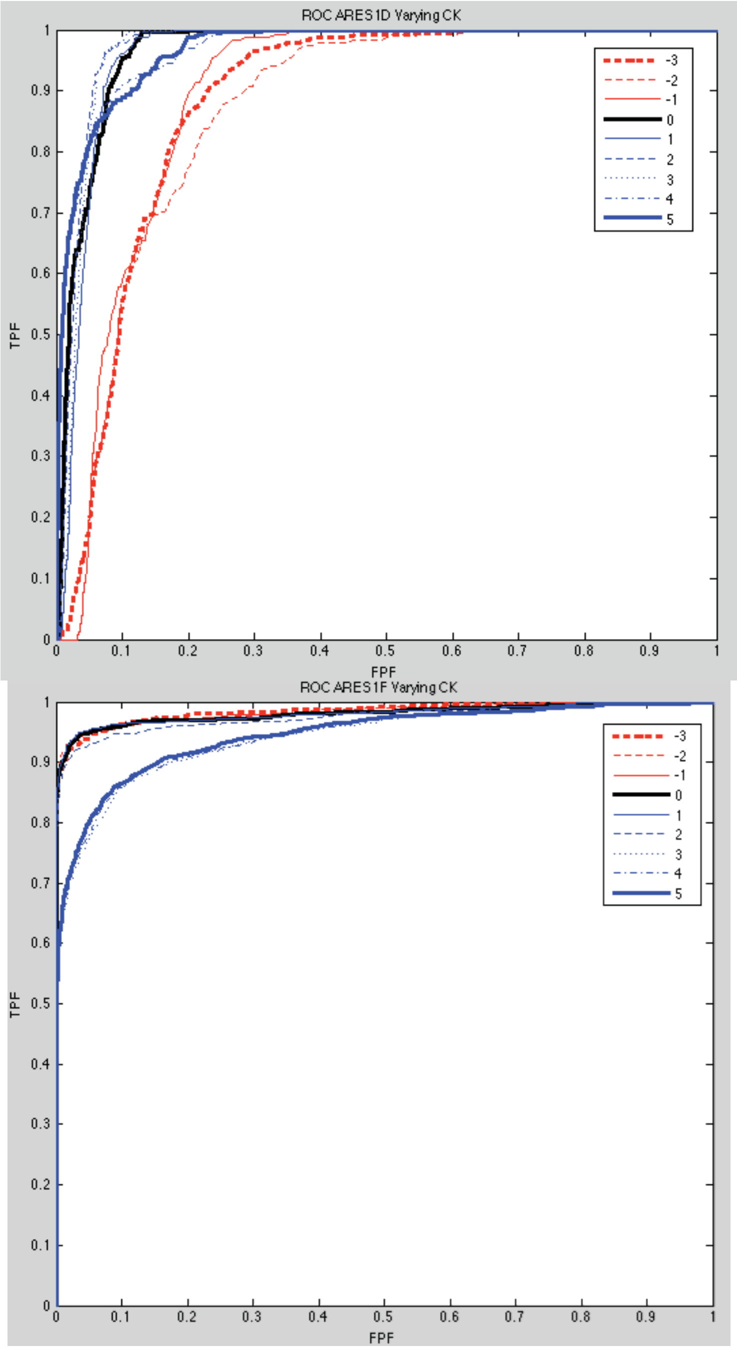


Figure 39: ROC Curves Varying Dimensionality Adjustments

The GIPREBAD algorithm can potentially amplify the effects of noise by removing noisy pixels that also contain background information. In the case of ARES1D, for instance, noisy pixels containing spectral information for the sagebrush present in the scene might be removed from the covariance estimation. The sagebrush is already sparsely present in the scene and its spectral signal diluted by shadows caused by the solar angle. These two factors might be causing the anomaly detector to become a "sagebrush detector" in this instance. One possible technique to avoid this problem would be to use the same adaptive filtering as in AutoGAD on the Q score such as to reduce the noise amplified by the iterative technique. Results on ARES1D and ARES1F after 20 IAN filtering iterations on the final Q scores yielded promising results as shown in Figure 40.

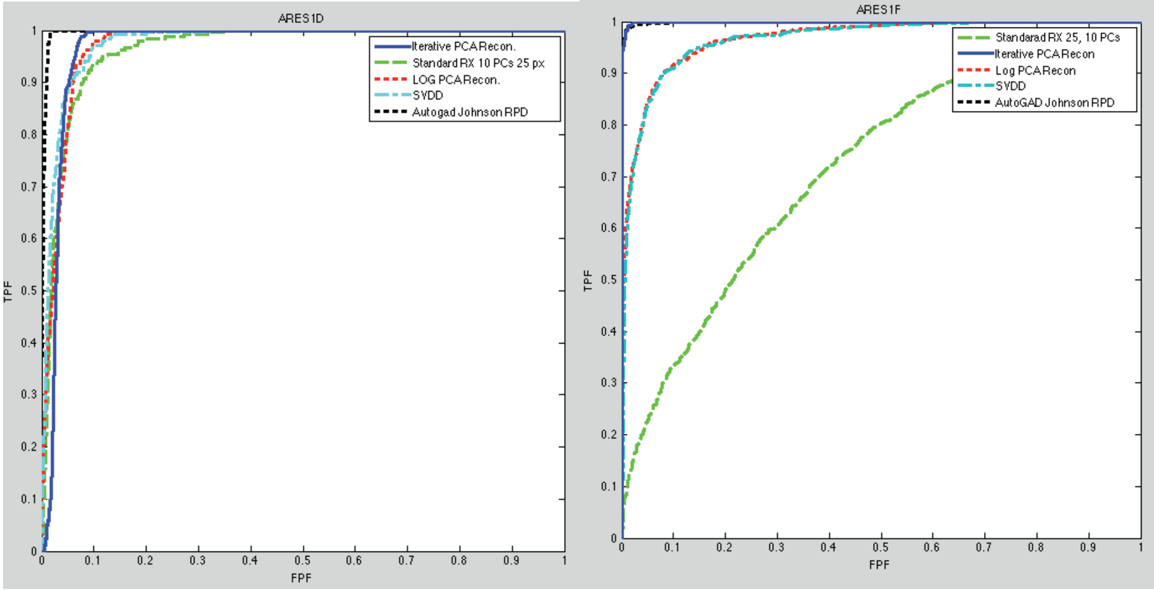


Figure 40: GIPREBAD with Q IAN Filtering

Interestingly, despite the impressive ROC curve generated by the adding of 20 iterations of IAN filtering, this is not necessarily a very desirable procedure. Filtering removes noise from the image, but adaptive filtering can also remove valid signal in the form of very small anomalies from the image after too many iterations. As an example, reference Figure 41. Here, it is apparent that the vehicles on the upper left of ARES1D in the filtered scene all but disappear. Furthermore, although there is far less noise in ARES1D when it is filtered, the tanks along the road lose some of their sharp geometric features, which would possibly make it harder for an analyst to declare/confirm that they are anomalies. In this way, care must be taken when judging an algorithm on TPF, FPF, and LA performance alone.

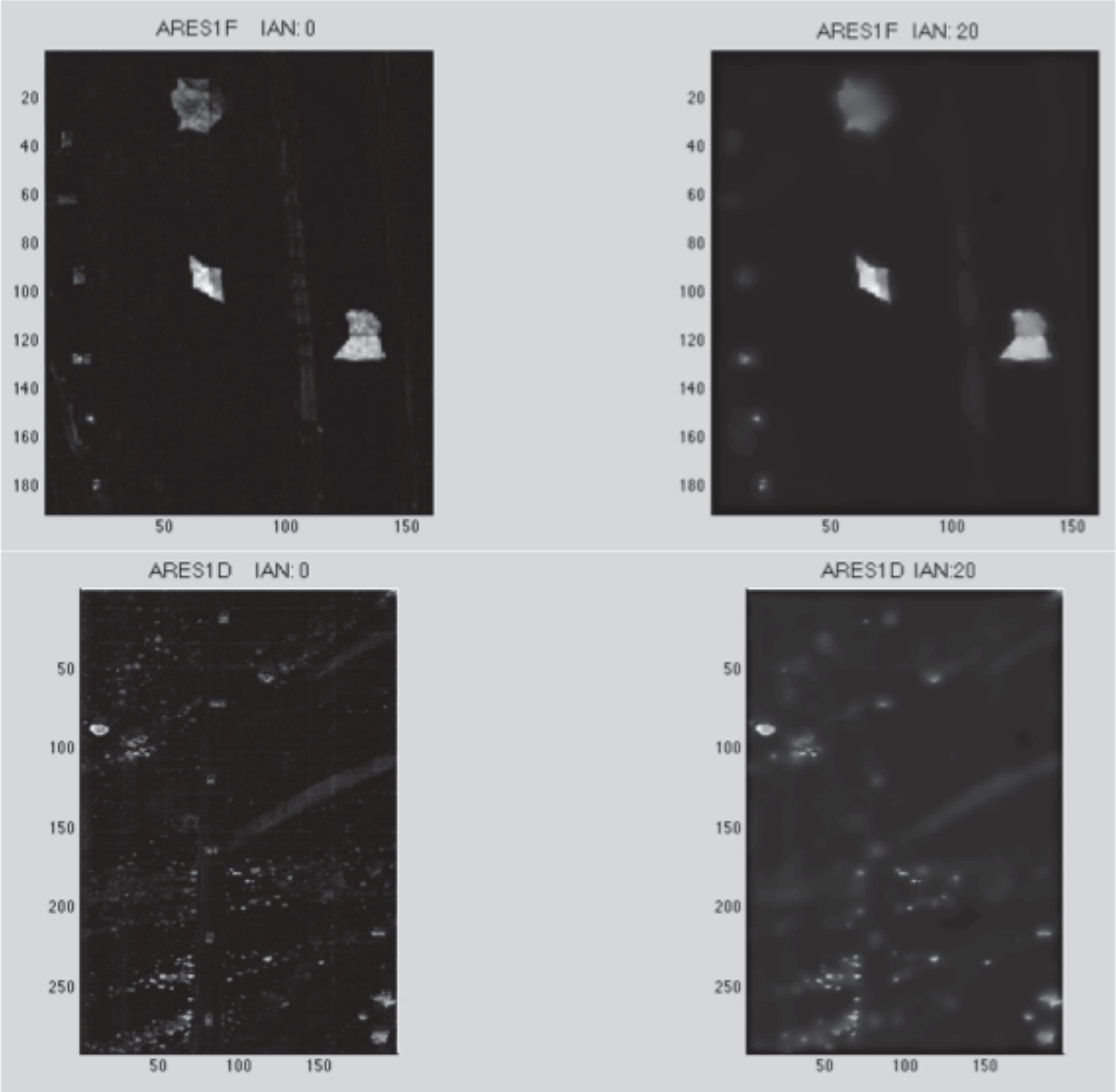


Figure 41: IAN Filtering & Final Abundance Maps

An exhaustive enumeration approach was taken to choose a group of settings for the GIPREBAD algorithm. A total of five parameters were considered, l PC filtering iterations, the detection sensitivity Y, the dimensionality adjustment c_k, the threshold for iterative PCA Q_{thresh}, and the maximum number of iterations. Every possible combination of values for the parameters and step sizes shown in Table 4 were visited for each image in the training set shown in Table 6 on page 94. The settings with the highest average area under the ROC curve were selected as the algorithm operating parameters. A test set is also included in Table 7, but the results will not be considered until Chapter IV.

Table 4: GIPREBAD Exhaustive Enumeration Limits

Parameter	Description	Bounds	Step Size
l	IAN filtering iterations	[0,12]	1
Y	Detection sensitivity	[0.5,3]	.25
c_k	Dimensionality adjustment	[-1,1]	1
Q_{thresh}	Iterative threshold	[1,4]	.2
maxiterations	Maximum iterations	[0,12]	1

The optimal combination of settings found is shown in

Table 5. Interestingly, the algorithm seemed to perform best with very few iterations and a very low Q_{thresh}. This seems to indicate that it is more effective to remove a large number of pixels from the data used for covariance estimation only a small number of times.

Table 5: GIPREBAD Exhaustive Enumeration Results

Parameter	Value
l	7
Y	.75
c_k	1
Q_{thresh}	1.4
$maxiterations$	2

Figure 42 shows the ROC curves for GIPREBAD when operating with the parameters determined through exhaustive enumeration. The dots printed along the curves indicate the actual operating point selected by the zero-detection histogram method. It seems as though a fairly robust set of parameters was selected as the result translates well to the test set. A more exhaustive examination of these results, as well as a performance comparison to other algorithms presented will be shown in Chapter IV.

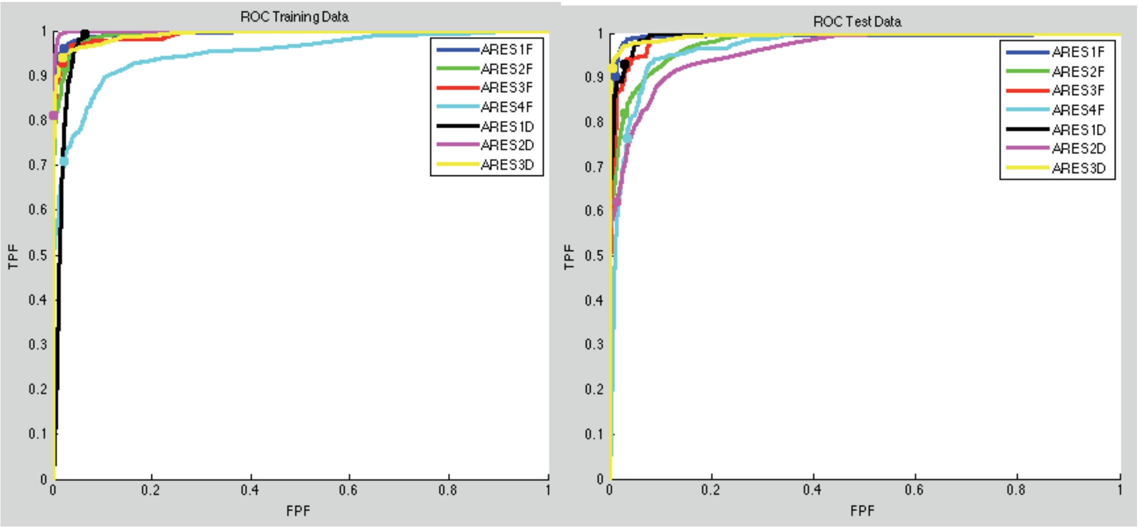
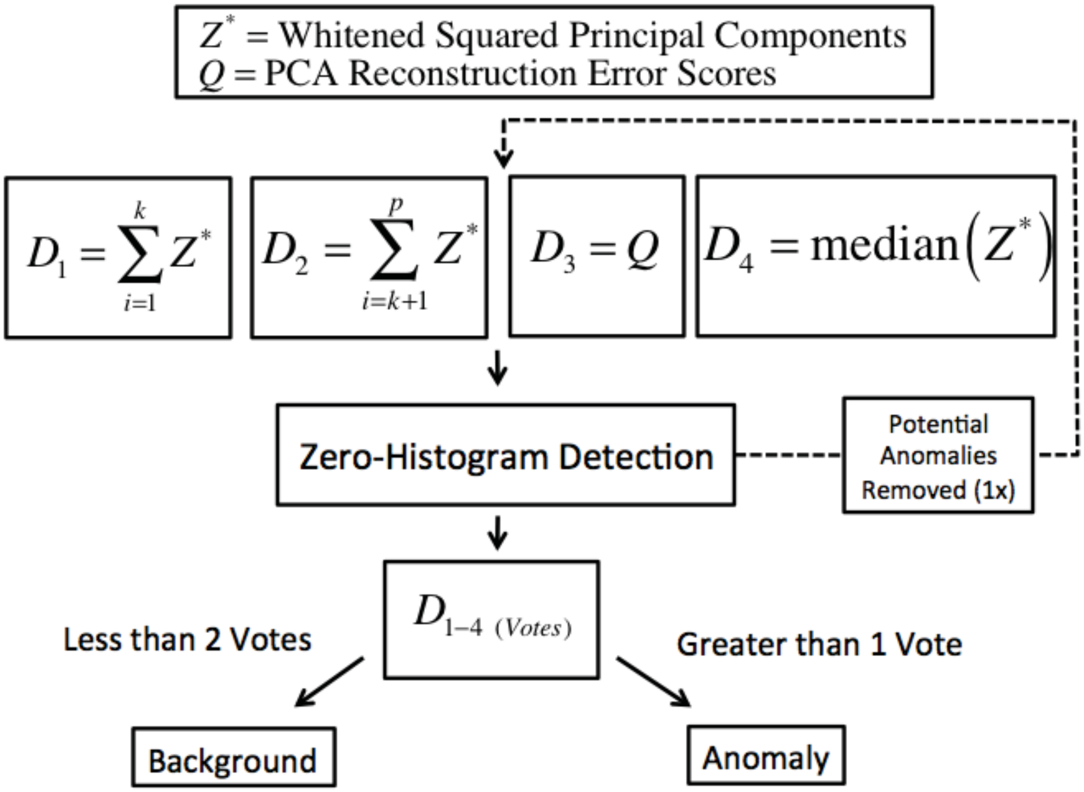


Figure 42: GIPREBAD Training and Test ROC Curves

In summary, the GIPREBAD algorithm is a fully autonomous global method for declaring anomalies within HSI. It repetitively ‘prunes’ the HSI data matrix used to estimate first and second order statistics used for PC construction. This enables a more accurate estimation of the background covariance structure and more accurate anomaly declarations using PCA reconstruction error. Even prior to optimization, with the two test images the algorithm performs competitively with the other standard HSI anomaly detectors discussed in Chapter 2. This section served to prove the effectiveness of PCA reconstruction error in anomaly detection, and illustrated the positive effect of obtaining ‘cleaner’ background covariance estimation prior to anomaly detection.

3.6 Multiple PCA

This section presents a fully autonomous global anomaly detector dubbed “Multiple PCA.” It consists of a voting ensemble that combines results of the zero-detection histogram method (19) on four PCA based anomaly detection scores, D_{1-4}, described later in this section. The voting scheme, shown in Figure 43, makes the detector more robust against the noise inherent in HSI, as well as to shortcomings of each individual score. Although the ensemble consists wholly of members using PCA based scores, they are diverse in their responses to different images and targets, which increases the ensemble’s effectiveness. Further, a highly sensitive initial anomaly detection is performed, and potential anomalous pixels are excluded during covariance estimation for the final anomaly declarations. As in GIPREBAD and Iterative RX, this serves to reduce anomaly effects on first and second order statistic estimations. The result is an algorithm that is highly robust to different images, conditions, and anomalies. Tangible improvements over AutoGAD and other standard detection algorithms are realized.

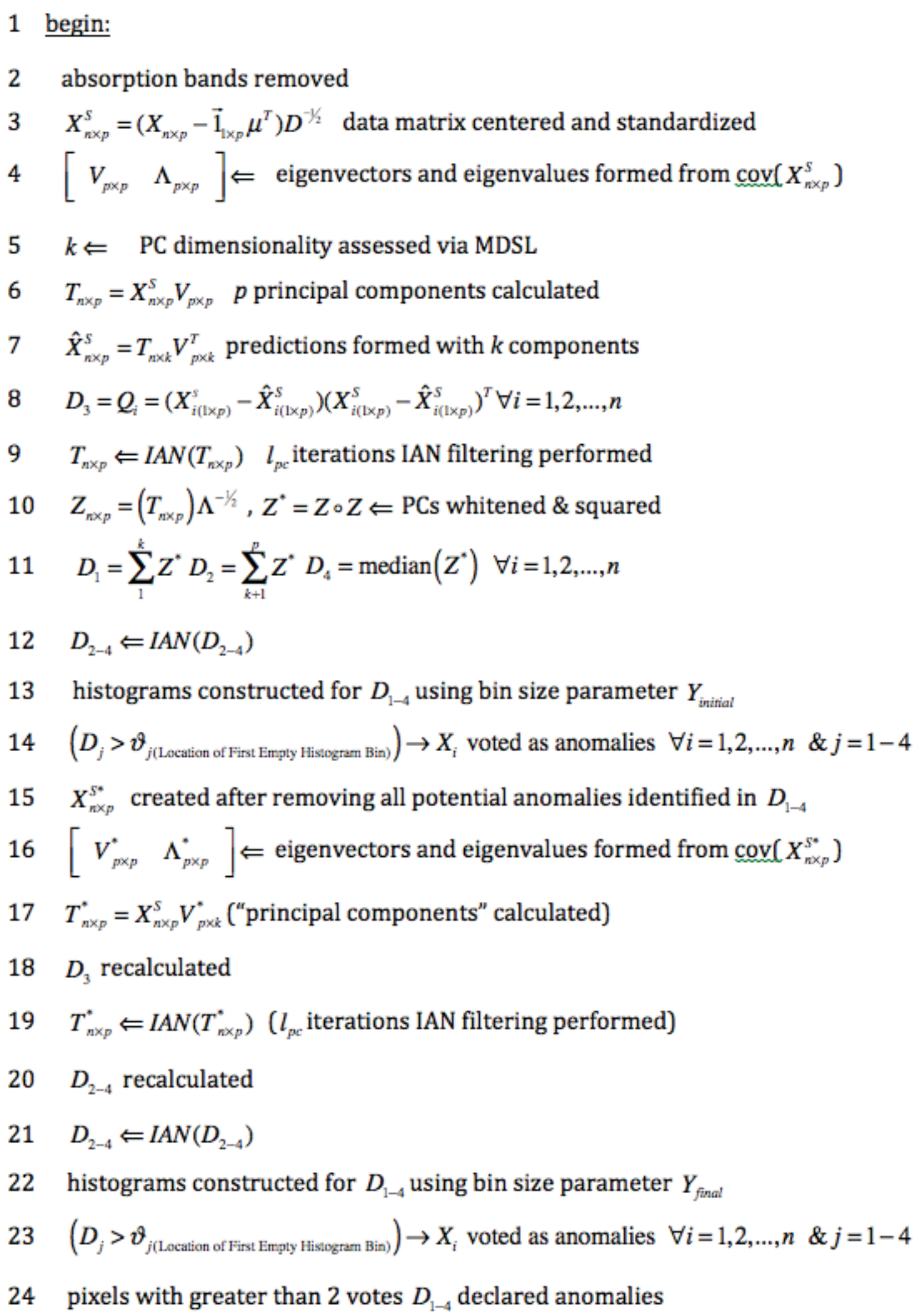


$Z^* =$ Whitened Squared Principal Components
$Q =$ PCA Reconstruction Error Scores
$D_1 = \sum_{i=1}^{k} Z^*$
$D_2 = \sum_{i=k+1}^{p} Z^*$
$D_3 = Q$
$D_4 = \text{median}\left(Z^*\right)$
Zero-Histogram Detection
Potential Anomalies Removed (1x)
D_{1-4} $(Votes)$
Less than 2 Votes
Greater than 1 Vote
Background
Anomaly

Figure 43: The Multiple PCA Algorithm

3.6.1 Algorithm Development

Pseudocode for Multiple PCA is provided in Figure 44 and will serve to frame the presentation of the algorithm. References to the pseudocode will be displayed as {line #}. The remainder of this section will cover data handling and the broad functioning of the algorithm {lines 2-5, 12-17, 21-24}. The next three sections will detail the formation of the D_{1-4} statistics using the images ARES1D and ARES1F. Finally, a response surface model and experimental design akin to RPD will be conducted to optimize the algorithm parameters.



```
1  begin:
2    absorption bands removed
3    $X^S_{n\times p} = (X_{n\times p} - \vec{1}_{1\times p}\mu^T)D^{-\frac{1}{2}}$   data matrix centered and standardized
4    $\begin{bmatrix} V_{p\times p} & \Lambda_{p\times p} \end{bmatrix} \Leftarrow$ eigenvectors and eigenvalues formed from cov( $X^S_{n\times p}$ )
5    $k \Leftarrow$   PC dimensionality assessed via MDSL
6    $T_{n\times p} = X^S_{n\times p} V_{p\times p}$   $p$ principal components calculated
7    $\hat{X}^S_{n\times p} = T_{n\times k} V^T_{p\times k}$   predictions formed with $k$ components
8    $D_3 = Q_i = (X^s_{i(1\times p)} - \hat{X}^S_{i(1\times p)})(X^S_{i(1\times p)} - \hat{X}^S_{i(1\times p)})^T \forall i = 1,2,\ldots,n$
9    $T_{n\times p} \Leftarrow IAN(T_{n\times p})$   $l_{pc}$ iterations IAN filtering performed
10   $Z_{n\times p} = \left(T_{n\times p}\right)\Lambda^{-\frac{1}{2}}$ , $Z^* = Z \circ Z \Leftarrow$ PCs whitened & squared
11   $D_1 = \sum_1^k Z^*\ D_2 = \sum_{k+1}^p Z^*\ D_4 = \text{median}\left(Z^*\right)\ \ \forall i = 1,2,\ldots,n$
12   $D_{2-4} \Leftarrow IAN(D_{2-4})$
13   histograms constructed for $D_{1-4}$ using bin size parameter $Y_{initial}$
14   $\left(D_j > \vartheta_{j(\text{Location of First Empty Histogram Bin})}\right) \rightarrow X_i$ voted as anomalies $\forall i = 1,2,\ldots,n$ & $j = 1-4$
15   $X^{S*}_{n\times p}$ created after removing all potential anomalies identified in $D_{1-4}$
16   $\begin{bmatrix} V^*_{p\times p} & \Lambda^*_{p\times p} \end{bmatrix} \Leftarrow$ eigenvectors and eigenvalues formed from cov( $X^{S*}_{n\times p}$ )
17   $T^*_{n\times p} = X^S_{n\times p} V^*_{p\times k}$ ("principal components" calculated)
18   $D_3$ recalculated
19   $T^*_{n\times p} \Leftarrow IAN(T^*_{n\times p})$ ( $l_{pc}$ iterations IAN filtering performed)
20   $D_{2-4}$ recalculated
21   $D_{2-4} \Leftarrow IAN(D_{2-4})$
22   histograms constructed for $D_{1-4}$ using bin size parameter $Y_{final}$
23   $\left(D_j > \vartheta_{j(\text{Location of First Empty Histogram Bin})}\right) \rightarrow X_i$ voted as anomalies $\forall i = 1,2,\ldots,n$ & $j = 1-4$
24   pixels with greater than 2 votes $D_{1-4}$ declared anomalies
```

Figure 44: Pseudocode for Multiple PCA

As in GIPREBAD, before PCA is performed, the absorption bands are removed and the data, $X_{n\times p}$, is centered and standardized to obtain $X^S_{n\times p}$ {line 3}. The dimensionality, k , of the data is then assessed via the maximum Euclidean distance from the log-scale secant line (MDSL) technique {line 5} (Johnson R. J., 2008). An adjustment factor, c_k , as presented in section 3.5 may be used to adjust for chronic over or underestimation of dimensionality to effect better algorithm performance, $k = k + c_k$, before continuing. After PCA is performed on, $X^S_{n\times p}$, the D_3 score is calculated using k principal components {line 8}. Subsequently, l_{pc} iterations of adaptive noise filtering (IAN) are performed on the trailing principal components ($k \to p$) {line 9}. As in AutoGAD, adaptive filtering is chosen as it filters more heavily in areas where the variance is close to system noise, while not heavily filtering areas with significant signal (Johnson R. J., 2008).

After filtering, the D_1 , D_2 , and D_3 scores are computed {line 11}, and additional IAN filtering with l_D iterations is performed on D_{2-4} {line 12}. Four separate histograms corresponding to each D score are then constructed with a highly sensitive bin size parameter $Y_{initial}$. The zero-detection histogram method is then used to declare potential anomalies {line 14}. PCA is again performed, but using covariance estimates not including the potential anomalies {line 14}. The D_3 statistic is then recalculated {line 15}, again followed by l_{pc} iterations of adaptive filtering on the trailing principal components {line 16}. The final D_2 , D_3 , and D_4 scores are then calculated {line 17}, followed once again by IAN filtering with l_D iterations on D_{2-4} {line 21}. Finally, another round of zero-detection for D_{1-4} , after histogram construction with a less sensitive bin-size parameter Y_{final} {line 23}, determines voting for each individual component. For final anomaly declaration, at least 2 votes are required to declare a pixel anomalous {line 24}.

3.6.2 *D₁* and *D₂*

The D_1 and D_2 statistics (Jolliffe, 2002) both require whitening of the principal components,

$$Z_{n\times p} = \left(T_{n\times p}\right)\Lambda^{-\frac{1}{2}} \tag{47}$$

Values in Z are then squared by element, $Z^* = Z \circ Z$, and the two scores are simply linear combinations of the result. D_1 is the sum the first k whitened squared components,

$$D_1 = \sum_{1}^{k} Z^* \tag{48}$$

while D_2 is the sum of the remaining components,

$$D_2 = \sum_{k+1}^{p} Z^* \tag{49}$$

The D_1 score or any detector that relies solely on the first k PCs serves to identify anomalies that would generally be observable by looking at plots of the original data or plots of the individual PCs. These anomalies inflate variances and covariances as they caused large increases in one or more of the variances in the original variables (Jolliffe, 2002). Theoretically, AutoGAD and other algorithms that focus solely on the first k principal components will be very good at picking out anomalies of this type.

As an example, recall how easily the vehicles along the road in ARES1D were visible from looking at plots of the first few principal components in Figure 6. Figure 45 shows the D_1 score for ARES1D, note there was only a handful of false negatives for the vehicles and no fringe pixels were declared anomalous. In the results for D_1 for ARES1F in Figure 45 there are similar results and one can note the outline of the tent features being false negatives. This might be due to spectral mixing causing these pixels, although truly anomalous, to have little effect on the overall covariance structure.

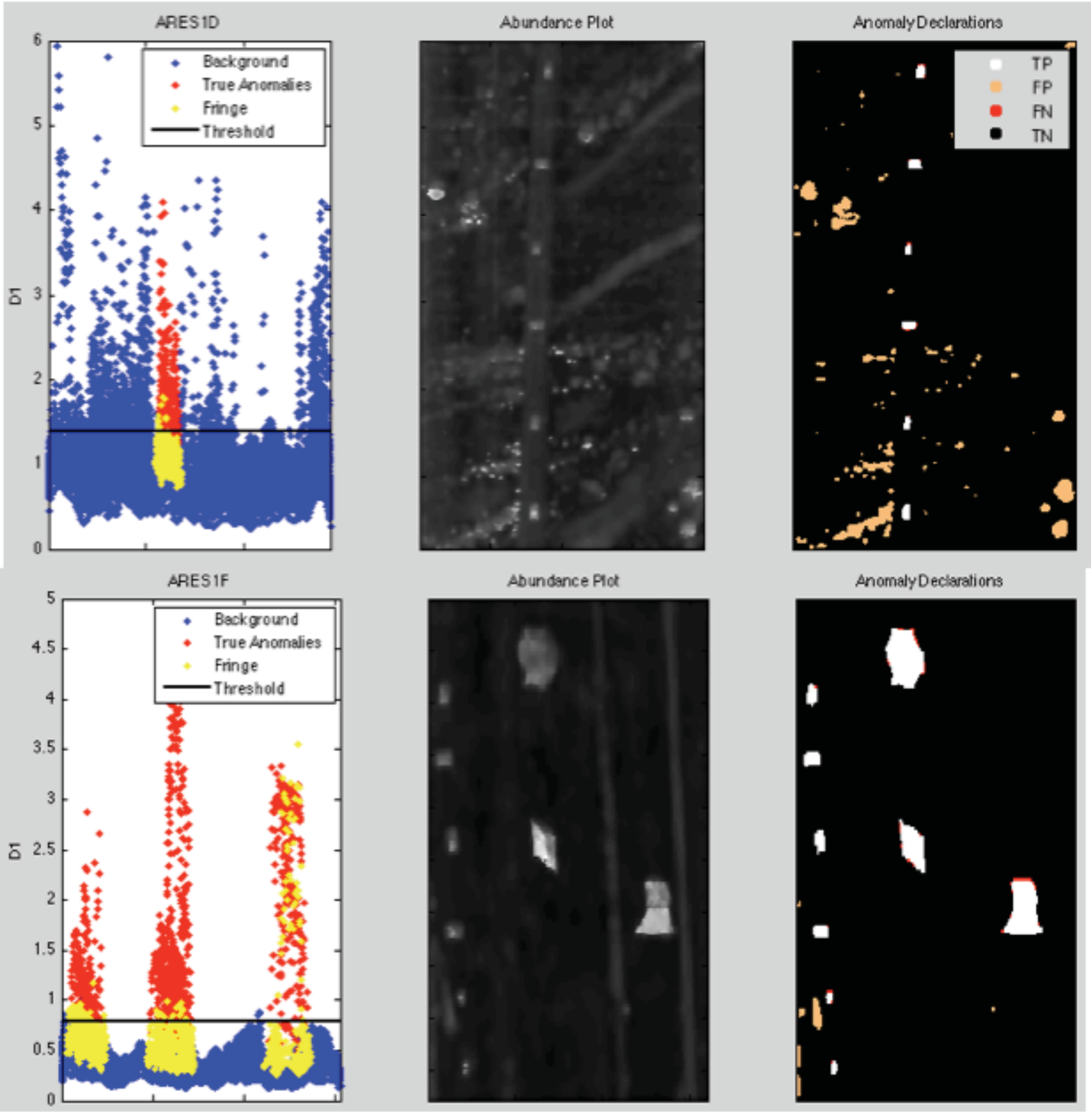


Figure 45: ARES1D & ARES1F D_1 Score

In contrast, the D_2 score may be able to detect outliers that are not apparent when looking at plots of the individual PCs or original bands. These band structures do not adhere to the overall covariance structure, but are not extreme enough in any one variable to affect the overall covariance estimates (Jolliffe, 2002). Although the effect is not very dramatic for the two test images, one can see the failure of D_2 to capture some of the more ‘obvious’ anomalies, while seeming to capture some of the fringe pixels around the vehicles in ARES1D.

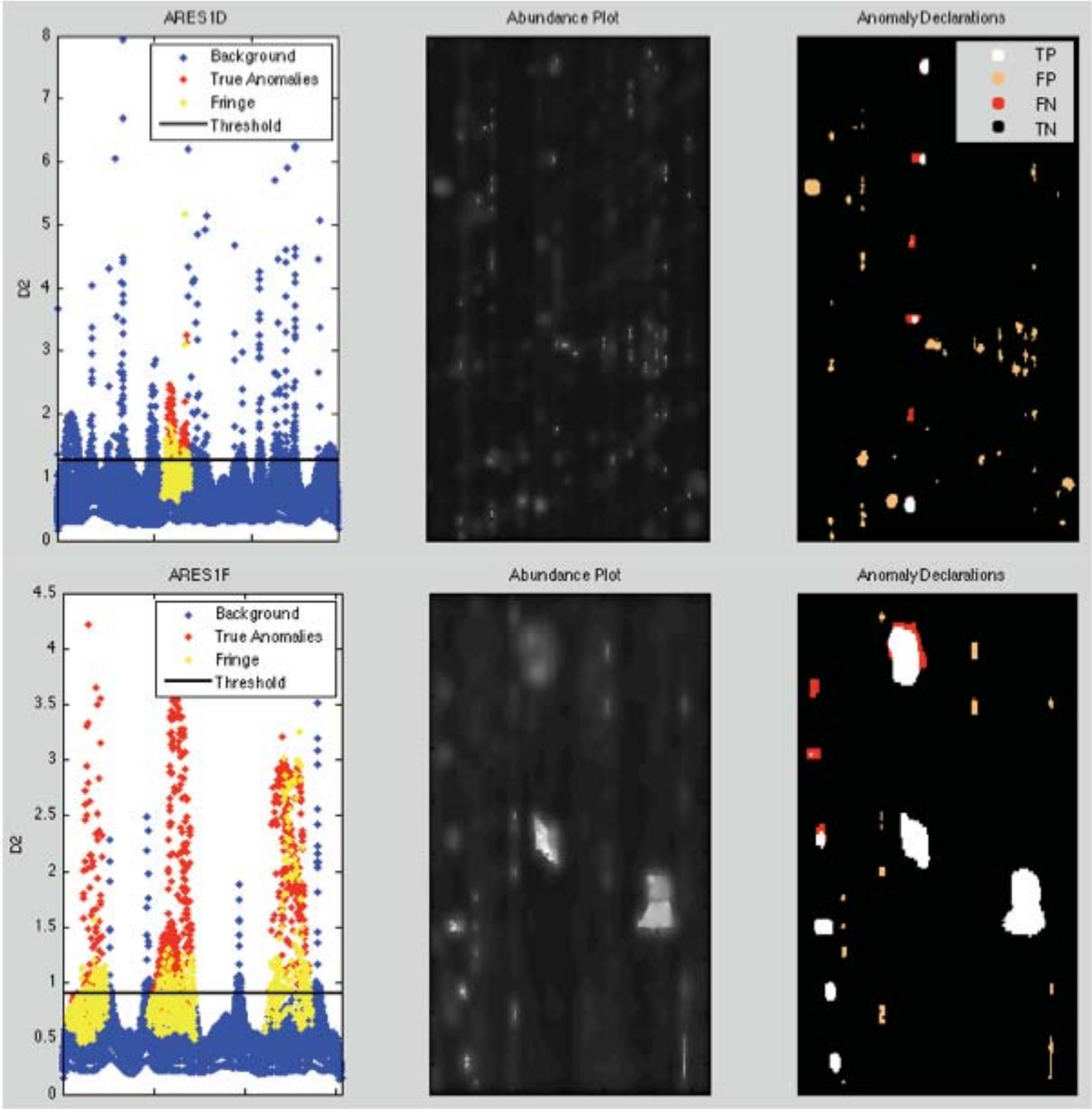


Figure 46: ARES1D & ARES1F D_2 Score

3.6.3 D_4

The D_4 statistic is the value of the median component of Z^* by pixel,

$$D_4 = \text{median}\left(Z_i^*\right) \forall i = 1,2,\ldots,n \tag{50}$$

The author introduced this novel score because of the large amount of noise usually present in the latter principal components. It was thought that the median would be less volatile to noise in a pixel than the sum of squares. It is interesting to note that the linear combination of across all whitened squared principal components is equivalent to the Mahalanobis distance (Jolliffe, 2002).

$$r_{MD}(X_i) = (X_i - \bar{X})^T S^{-1} (X_i - \bar{X})\,. \tag{51}$$

This is readily seen when considering

$$S = V \Lambda V^T,\;\; T_{n\times p} = X_{n\times p} V_{p\times p},\;\; \&\;\; X = T_{n\times p} V^T_{p\times p}\,. \tag{52}$$

A simple proof is provided:

$$\begin{aligned} (X - \bar{X})^T S^{-1} (X - \bar{X}) &= (T_i - \mu_T) V^T V \Lambda^{-2} V^T V (T_i - \mu_T) \\ &= (T_i - \mu_T)^T \Lambda^{-2} (T_i - \mu_T) \\ &= \sum_{j=1}^{p} \frac{T_{ij}^2}{\lambda_j} \forall i = 1, 2, \ldots, p \end{aligned} \tag{53}$$

In this way, the D_4 score is a robust estimator to this measure. The score seems to capture anomalies that fall into both D_1 and D_2 and thus serves as a good member of the ensemble, especially with a two vote requirement.

Results for the two test images are shown in Figure 47, where one can note the slight decrease in noise levels in this score. In ARES1F, the high variability and average scores of the anomalies in relation to the rest of the image is very apparent. It provides a cleaner and seemingly more accurate anomaly detection than is shown in the previous three scores for both images. Noisy late trailing components, and background structure bearing early components are at least partially ignored by this score leading to its high performance as an anomaly detector in HSI.

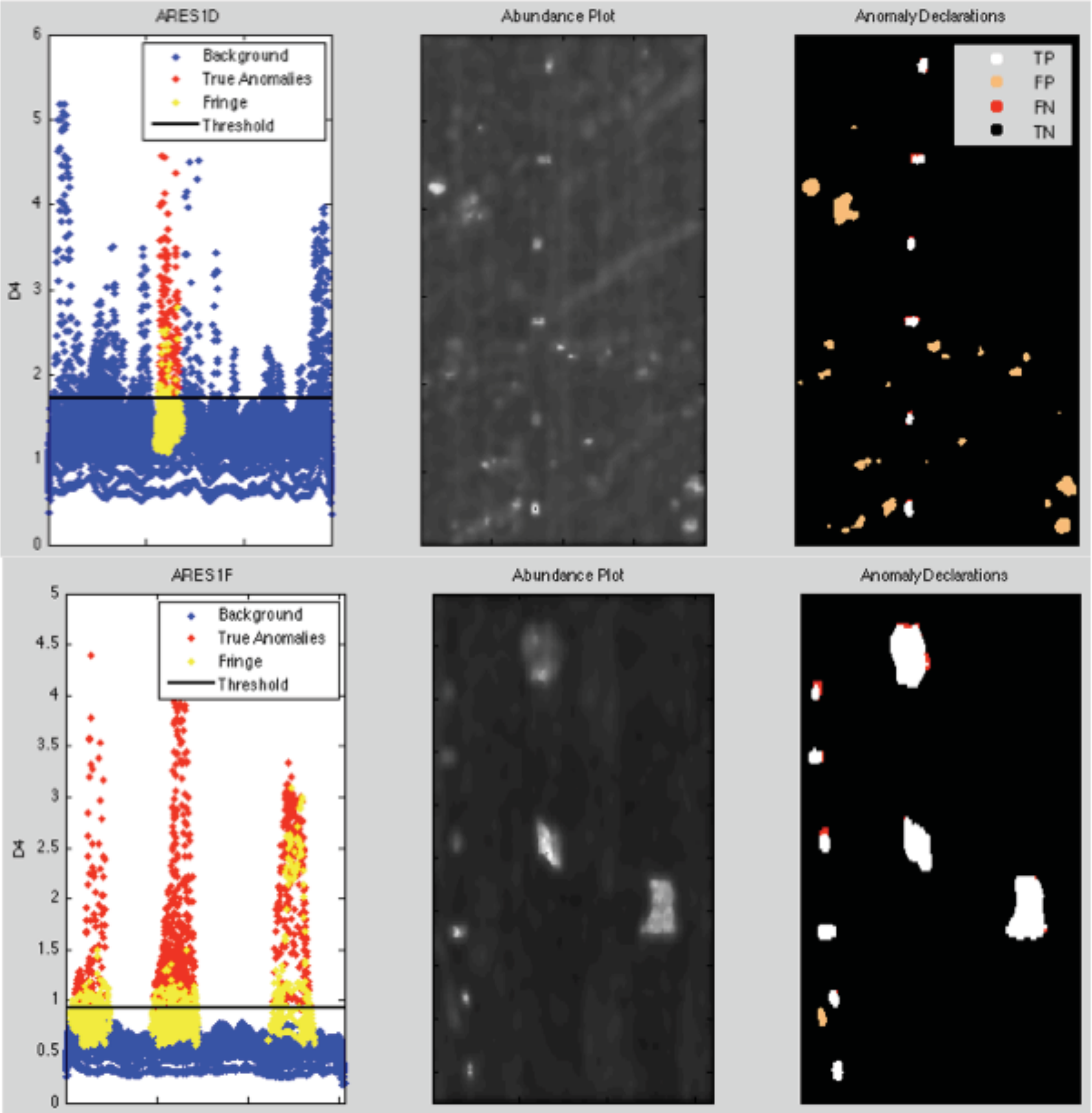


Figure 47: ARES1D and ARES1F *D*$_4$ Statistic

3.6.4 D$_3$

The D_3 score is the same as the, Q, or squared reconstruction error score used in the GIPREBAD and Log PCA reconstruction algorithms.

$$D_3 = Q_i = (X^s_{i(1\times p)} - \hat{X}^S_{i(1\times p)})(X^S_{i(1\times p)} - \hat{X}^S_{i(1\times p)})^T \forall i = 1,2,\ldots,n \tag{54}$$

It should be noted that this is a similar score to that of D_2 except that it does not give equal weighting to all of the principal component directions through whitening.

Recalling section 2.5.1, consider that the reconstruction error is equal to the linear combination of the *trailing* principal components

$$T_{i(1\times p-k)}V^{T}_{p\times p-k} = (X^{S}_{i(1\times p)} - \hat{X}^{S}_{i(1\times p)})\forall i = 1,2,\ldots,n. \tag{55}$$

In this way it complements the other statistics in the ensemble well by providing what might be considered weighted information from the trailing PCs. Earlier principal components contain more variance and are usually of greater magnitude, this aspect theoretically minimizes the noise that is prevalent in the later PCs.

Figure 48 shows results for the algorithm on the two algorithm development images. Noise is clearly a problem in ARES1D , while the statistic does extremely well in ARES1F. The problems in ARES1D are most likely due to relatively early PC bands being dominated by noise and sensor artifacts. The linear artifacts present in the Figure 6 appear in this score's abundance plots. It should also be noted that the noise present in the D_3 score is markedly different than that shown by the other scores. This, once again, is an indicator of its utility as a part of the multiple PCA ensemble.

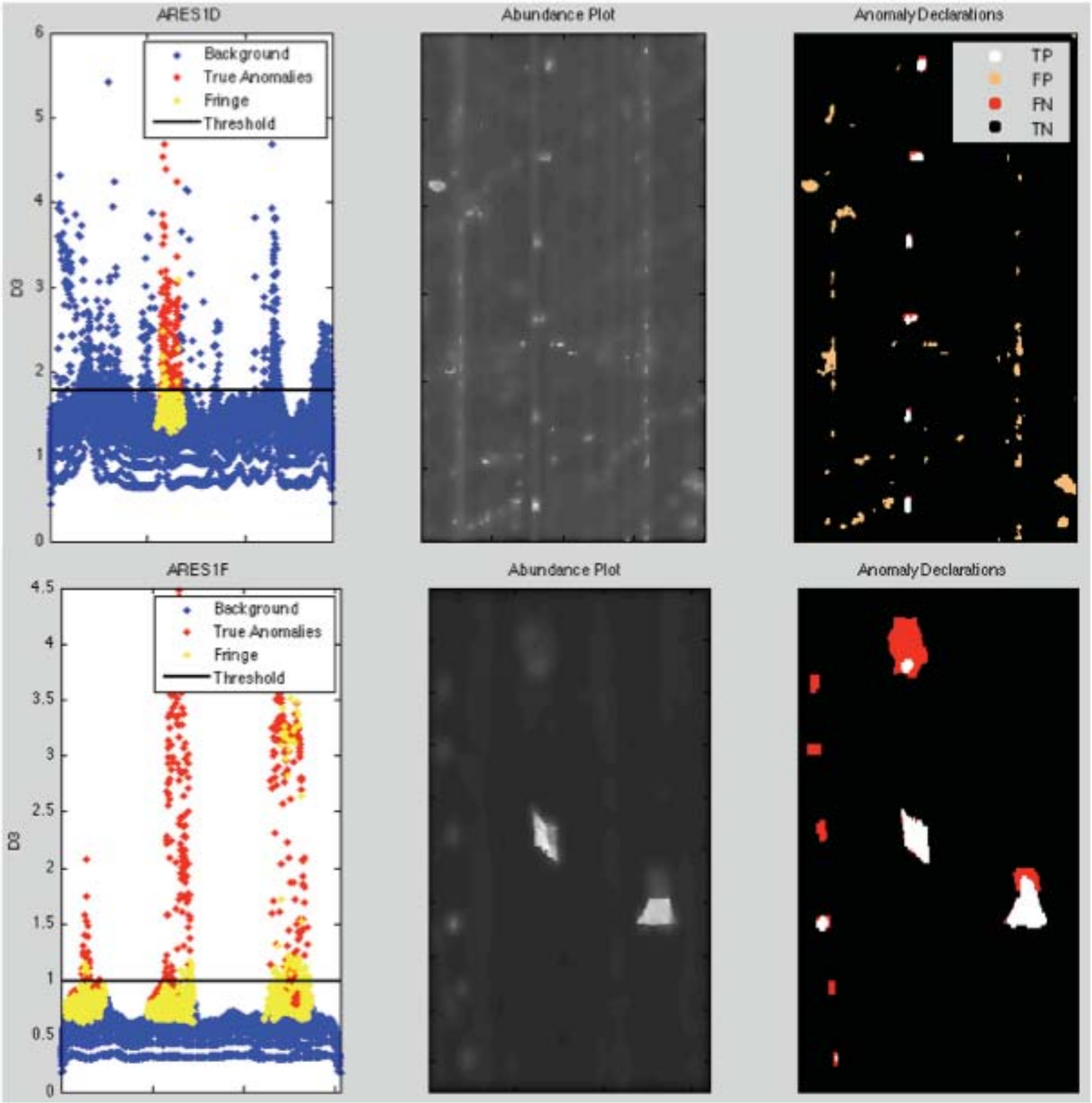


Figure 48:ARES1D and ARES1F D_3 Statistic

3.6.5 Summary of Technique

An appropriately constructed fusion of detectors leverages the strengths of individual members, while also masking some of their weaknesses. The Multiple PCA's voting ensemble does just this by combining four diverse scores in a voting ensemble. Using PCA as a basis for the four scores enables efficiency, as each score is simply different linear combinations of the original PCs. The exclusion of potential outliers in the background covariance estimation used for final anomaly detection makes the algorithm more robust against extremely anomalous pixel spectra and noise. Results for the algorithm on ARES1D and ARES1F are shown in Figure 49. Here we see a two very clean sets of anomaly declarations, especially compared to the individual score declarations above.

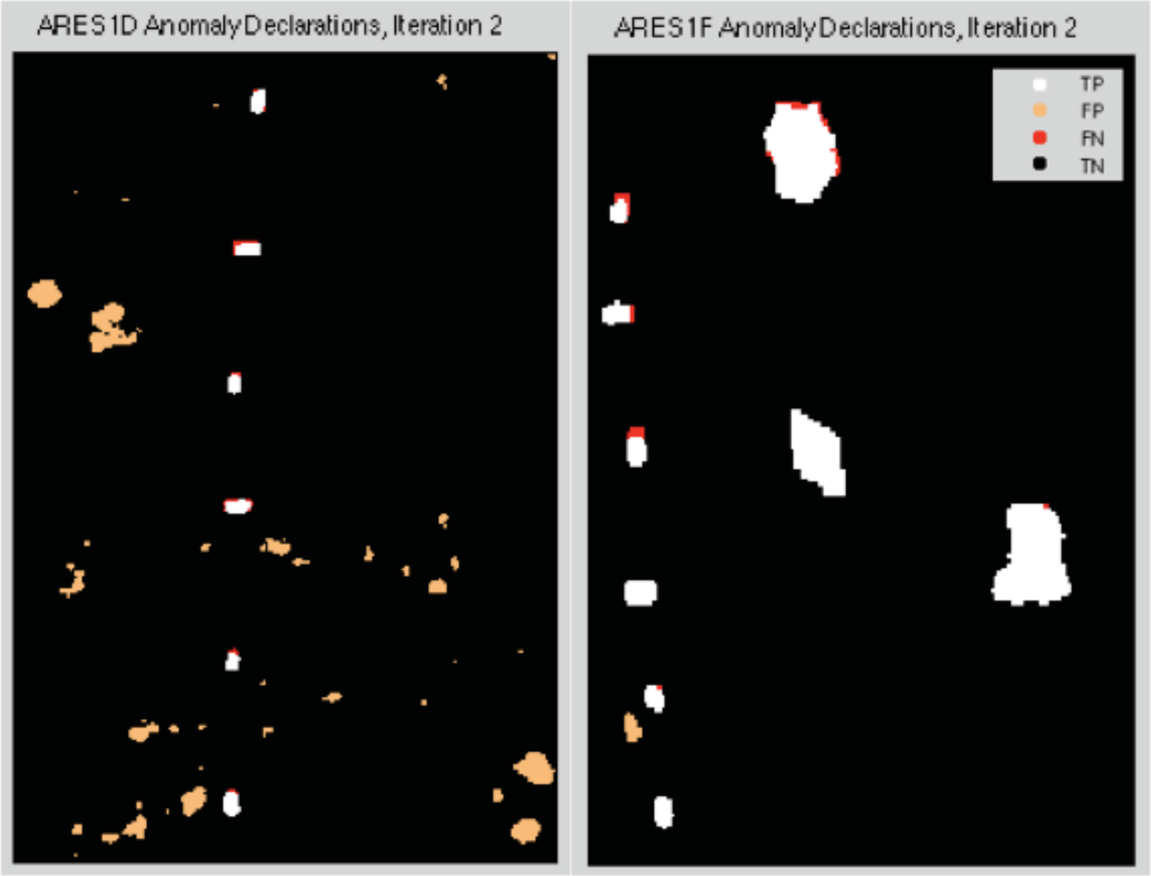


Figure 49: ARES1D & ARES1F Finaly Anomaly Declarations

3.6.6 Parameter Optimization

3.6.6.1 Purpose

Initial algorithm testing was largely accomplished by trial and error, and an original solution for algorithm parameter settings was determined through what might be considered subjectively directed exploration. Perhaps not unsurprisingly, this resulted in a poor translation of performance from training to test images and revealed the need for a systematic method of finding optimal algorithm parameters. The ROC curves in Figure 50 illustrate this well. Here, the solid lines indicate the detection capability after the removal of potential anomalies, while the dashed lines indicate detection results when using the entire image to estimate covariance for PCA.

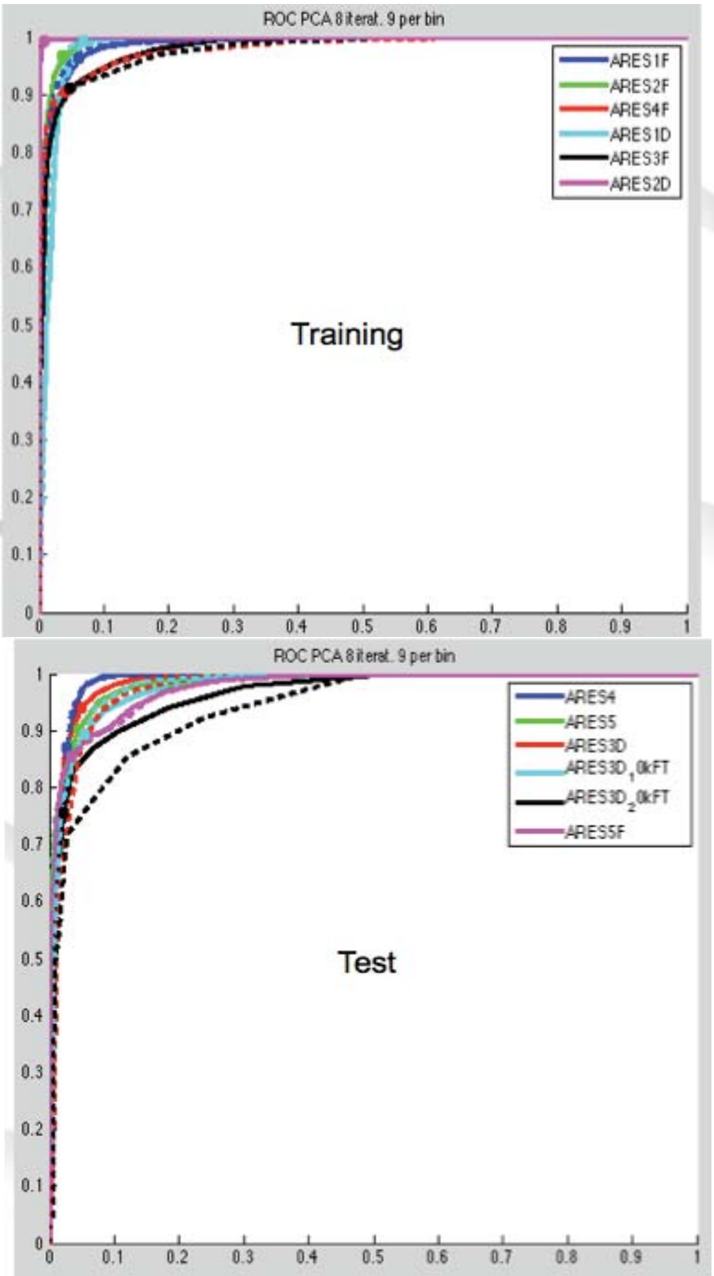


Figure 50: Multiple PCA Initial Results

Two things can be gathered from Figure 50, one is the increase in performance after removal of potential anomalies. The other is the poor performance of the algorithm with the original settings on the test set. Clearly, the receiver operating characteristics for the test images vary dramatically and indicate far less detection capability than that of the training set. The cause of this is twofold. First, the training and test sets were not chosen carefully for the images. Second, the settings developed during initial experimentation were not chosen systematically. For this reason, the difference in performance is also likely the result of ‘lucky overtraining,’ or happening upon a solution that worked very well for the training set but not for the test images. To address the aforementioned problems a training and test set was purposefully chosen, and a response surface method akin to RPD was used to optimize the Multiple PCA algorithm. These two steps prevented overtraining while ensuring accurate anomaly detection across a diverse set of images.

3.6.6.2 Training and Test Set Construction

A new training set was constructed focusing on diversity of training examples and is shown in Table 6. The set of images contains three desert and four forest scenes, divided evenly to encourage a consistent response between scene types. Three images ARES2F, ARES4F, and ARES1D contain extremely small concentrations of target pixels; the rest of the images span a range up to the maximum concentration of target pixels available in the HYDICE data available. The Fisher ratio (Duda, Hart, & Stork, 2001), F_{ratio}, calculated

$$F_{ratio} = \frac{\sum_{i=1}^{p}\left(\frac{\left(\mu_{ai}-\mu_{bi}\right)^2}{\sigma_{ai}^2+\sigma_{bi}^2}\right)}{p}, \tag{56}$$

where μ_{ai} and μ_{bi} are the anomaly and background band means, σ_{ai}^2 and σ_{bi}^2 are the anomaly and background band variances, was also considered. The Fisher ratio measures the discriminating power of a variable, and therefore provides an estimation of the ease with which an anomaly detector can discern the anomalies from the background. Furthermore, as sample size can affect both the zero-detection histogram method and the effects of outliers on covariance estimation, a substantial range of image sizes was also selected.

Table 6: Training Images

Image	Pixels	Dim.	% Target Pixels	Targets	Fisher Ratio	Scene
ARES1F	30560	191x160	3.30%	10	39.87	Forest
ARES2F	47424	312x152	0.65%	30	125.68	Forest
ARES3F	30736	226x136	1.02%	20	34.60	Forest
ARES4F	16400	205x80	0.66%	29	5.13	Forest
ARES1D	57909	291x199	0.41%	6	112.45	Desert
ARES2D	22360	215x104	2.34%	46	6.99	Desert
ARES3D	24336	156x156	1.80%	4	72.81	Desert

The test set was chosen with a similar mentality. First, the seven images are divided evenly between forest and desert scenes. Second, there is a large range of target pixel concentrations and the Fisher ratio varies similarly to the training set. Finally, a few very small images are included, as well as the largest image available to test the detection algorithm’s abilities to handle varying image sizes. The test set is shown in Table 7. A validation set with two images that contain no anomalies will be used to confirm algorithm accuracy as well and is displayed in Table 8.

Table 7: Test Images

Image	Pixels	Dim.	% Target Pixles	Targets	Fisher Ratio	Scene
ARES4	35880	460x78	2.46%	15	61.13	Desert
ARES5	53250	355x150	1.10%	15	28.46	Forest
ARES5F	72850	470x155	1.48%	45	19.62	Forest
ARES3D_10kFT	9450	139x68	1.37%	28	84.28	Desert
ARES3D_20kFT	4453	61x73	1.15%	4	129.67	Desert
ARES6D_10kFT	16555	215x77	0.87%	13	37.41	Desert
ARES7F_10kFT	14168	161x88	2.71%	12	17.50	Forest

Table 8: Validation Images

Image	Pixels	Dim.	% Target Pixles	Targets	Fisher Ratio	Scene
ARES1C	21924	203x108	n/a	0	n/a	Forest
ARES2C	24552	124x198	n/a	0	n/a	Forest

3.6.6.3 Optimization Function and Training Method

Measuring the utility of an anomaly detector requires a single performance metric that promotes both a high anomaly detection rate as well as a low false alarm rate. It is, of course, easy to achieve a high detection rate when false positives aren't very problematic. In the case of hyperspectral anomaly detection, it is important to keep the false alarm rate low so that an analyst or computer does not become overwhelmed with targets. In many anomaly detection situations this is also the case, and so any method to optimize anomaly detection algorithms must consider limiting its sensitivity and promoting accuracy.

The performance metric, P, chosen for algorithm optimization in this research was essentially a utility function with weightings selected to achieve a low consistent false alarm rate. The function took the form

$$P = \left(\mu_{TPF} - 1\right)^2 + 3\mu_{FPF}^2 + 3\sigma_{FPF}^2 . \tag{57}$$

μ_{TPF} , μ_{FPF} , and σ_{FPF}^2 are defined as

$$\mu_{TPF} = \frac{\sum_{i=1}^{N} TPF_i}{N} , \tag{58}$$

$$\mu_{FPF} = \frac{\sum_{i=1}^{N} FPF_i}{N} , \tag{59}$$

and

$$\sigma_{FPF}^2 = \frac{\sum_{i=1}^{N} \left(FPF_i - \mu_{FPF}\right)^2}{N-1} \tag{60}$$

where TPF_i and FPF_i are the true positive fraction and false positive fraction for detection results on an individual image within the training set of size, N. In this way, μ_{TPF} and μ_{FPF} , are the mean TPF and FPF and σ_{FPF}^2 is the variance of the FPF across the set of training images. The performance, P is then calculated for each row in the design matrix. Both of the algorithms were optimized using a three level full factorial design, 3^5 with a corresponding 243 “batches” of seven test images.

The P utility score is similar to that of the Lin and Tu (Lin & Tu, 1995) model presented in (28), as it includes the variance of the output in the response and generates a single score for optimization. The FPF was weighted more heavily in order to make the results more usable in HSI analysis. Experience with the HYDICE data showed general inconsistency from image to image on TPF rate for varying algorithms. In this way, not including the variance of the TPF in the utility function prevents the optimization function from “desiring” consistent results, which might not be possible or would force the optimization settings to decrease the TPF rate on ‘easier’ images.

A second order response surface model was chosen, as it has been found to be useful in a variety of situations so long as the range of the control variables is appropriate (Myers, Montgomery, & Anderson-Cook, 2009). The model takes the form,

$$P = \beta_0 + x^T \beta_1 + x^T \beta_2 x + \varepsilon, \tag{61}$$

where β_0 is the intercept, x is a vector of parameter settings, β_1 is vector of control variable coefficients, β_2 is a matrix containing the quadratic control variable coefficients, and ε represents the error. The error term is quite complex as it accounts for variance added by many factors that will likely not be explained by the model, including the varied responses of the algorithm to each of the individual images in the training set.

The next two sections outline design of two experiments using the response surface model described to optimize the algorithm. These designs are both ultimately the result of sequential experimentation to find an appropriate set of control variables and their corresponding testing limits to both enable reasonable regression statistics and optimal statistics. The first optimization groups sets a uniform detection sensitivity, Y_{final}, for all the D scores and focuses on mainly on other algorithm parameters. The second optimization considers selecting optimal parameters of Y_{final} for each individual anomaly detection score.

3.6.6.4 Optimization I

The first optimization presented on the Multiple PCA algorithm was designed to find optimal values for the number of score filtering iterations l_D, the optimal number of

PC filtering iterations l_{pc}, the optimal dimensionality adjustment c_k, and the optimal initial and final detection sensitivities $Y_{initial}$ and Y_{final}. The corresponding ranges of these control variables are shown in Table 9.

Table 9: Optimization I Parameter Ranges

Control Variable	Description	Lower Limit	Upper Limit
l_D	D_{2-4} IAN filtering iterations	4	12
Y_{final}	Final detection sensitivity	1.75	3.75
c_k	Dimensionality adjustment	-4	-2
l_{pc}	PC IAN filtering iterations	$.25l_d$	$.75l_d$
$Y_{initial}$	Initial detection sensitivity	0.01	0.15

A response surface model was estimated after experimentation with a 3^5 full factorial design. The 243 different values for the performance metric, P, were fit to a second order polynomial model (61). The regression resulted in an r-squared value of .7351 (r-squared adjusted .7112), indicating that the model explained over 73% of the variance in the output performance metric as the control variables were adjusted. Only a small departure from normality in the residuals is indicated as shown by the normal probability plot and the residual vs. predicted plot in Figure 51. A slight departure from normality is to be expected, especially towards the lower end of the prediction range. This is because the performance metric chosen is bounded and cannot be negative.

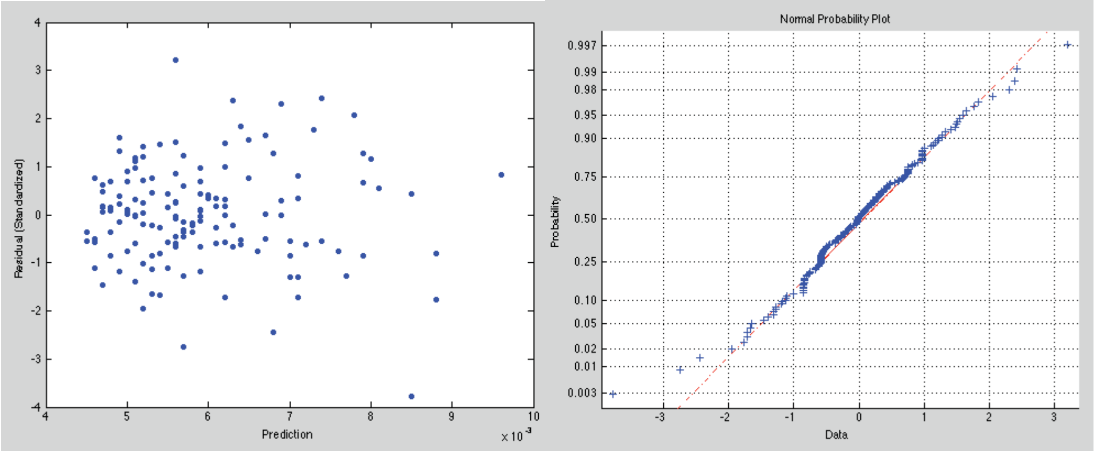


Figure 51: Optimization I Residual Analysis

A generalized reduced gradient algorithm was then implemented using the complete fitted model in order to minimize the output. The resultant solution revealed the control variables shown in Table 10. The solution was then applied to the training and test sets.

Table 10: Optimization I Parameter Results

Control Variable	Value
l_D	8
Y_{final}	2.775
c_k	-4
l_{pc}	$.25l_d$
$Y_{initial}$	0.142

Note the drastic difference between the results of the response surface model presented in Figure 52 and that of the originally selected parameters shown in Figure 50. There is clearly more consistency in the shape of the ROC curves as well as in the position of the operating points. The test set displays similar performance indicating that the parameters chosen are robust and can be applied successfully outside of the training set. A more detailed numerical analysis of these results will be offered in Chapter IV.

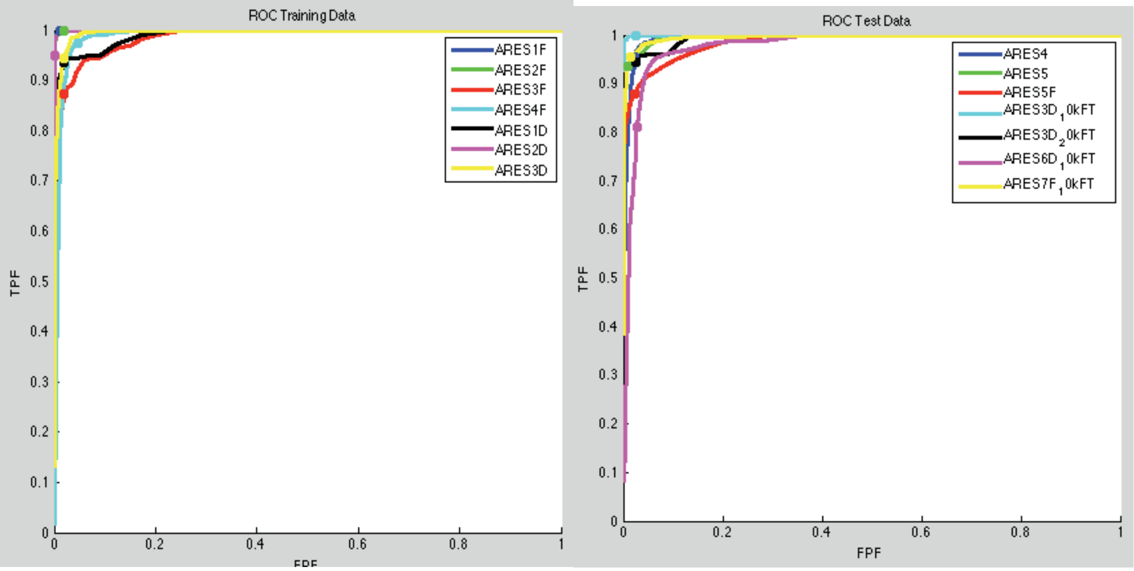


Figure 52: ROC Optimization I Results

3.6.6.5 Optimization II

The second optimization focused on selecting individual detection sensitivities $Y_{final\ D_{1-4}}$ for each score, and a corresponding, $Y_{initial}$, overall initial detection sensitivity. It was thought that due to the varying strengths between members of the ensemble, it would prudent to set individual zero-detection sensitivities for each of the four scores.   The settings obtained from the previous response surfaces results were used for the other settings in the algorithm. The control variables optimized and their associated limits are shown in Table 11.

Table 11: Optimization II Parameter Ranges

Control Variable	Description	Lower Limit	Upper Limit
$Y_{final\ D_1}$	D_1 Detection Sensitivity	1.5	3.5
$Y_{final\ D_2}$	D_2 Detection Sensitivity	1.5	3.5
$Y_{final\ D_3}$	D_3 Detection Sensitivity	1.5	3.5
$Y_{final\ D_4}$	D_4 Detection Sensitivity	1.5	3.5
$Y_{initial}$	Initial detection sensitivity	0.05	0.25

As before, a response surface model was constructed through experimentation with a 3^5 full factorial design. The 243 different values for the performance metric, P , were fit to a second order polynomial model (61). The regression resulted in an r-squared value of .8575 (r-squared adjusted .845), indicating that the model explained almost 86% of the variance in performance metric across settings. Furthermore, referencing the normal probability plot and the residual vs. predicted plot in Figure 53, one sees a slight problem with homoscedasticity and a moderate departure from normality. This can be considered a positive result, for as mentioned earlier, all of the statistics used to construct our performance metric are bounded.

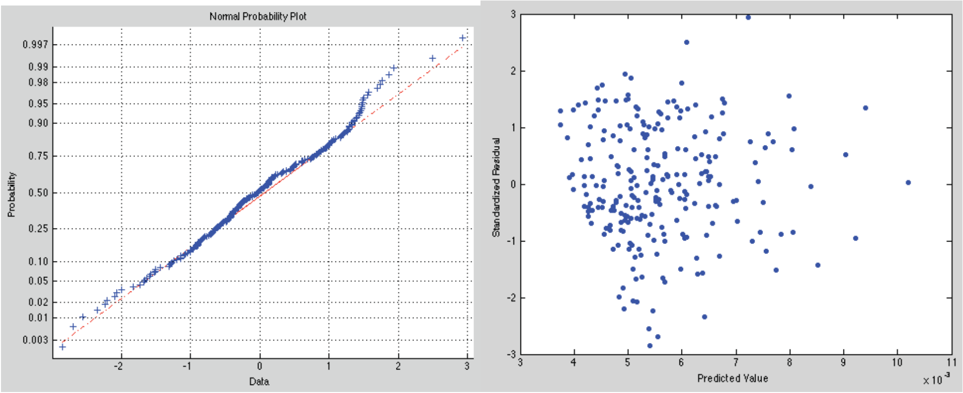


Figure 53: Optimization II Residual Analysis

Using the entire fitted model with all interaction terms, a generalized reduced gradient algorithm was again used to minimize the output. The resultant solution revealed the control variables shown in Table 12. The solution was again applied to the training and test sets.

Table 12: Optimization II Parameter Ranges

Control Variable	Value
$Y_{final\ D_1}$	3.500
$Y_{final\ D_2}$	2.774
$Y_{final\ D_3}$	2.856
$Y_{final\ D_4}$	2.295
$Y_{initial}$	0.249

The receiver operating characteristics of Multiple PCA with the optimized parameters readily show the utility of the response surface method. They are displayed in Figure 54. Once again, there is a notable consistency between the training and test set and the operating points remain very consistent. It seems as though optimizing the algorithm sensitivity for each score also showed some improvement in the ROC's as well. A detailed numerical comparison will be presented in Chapter IV.

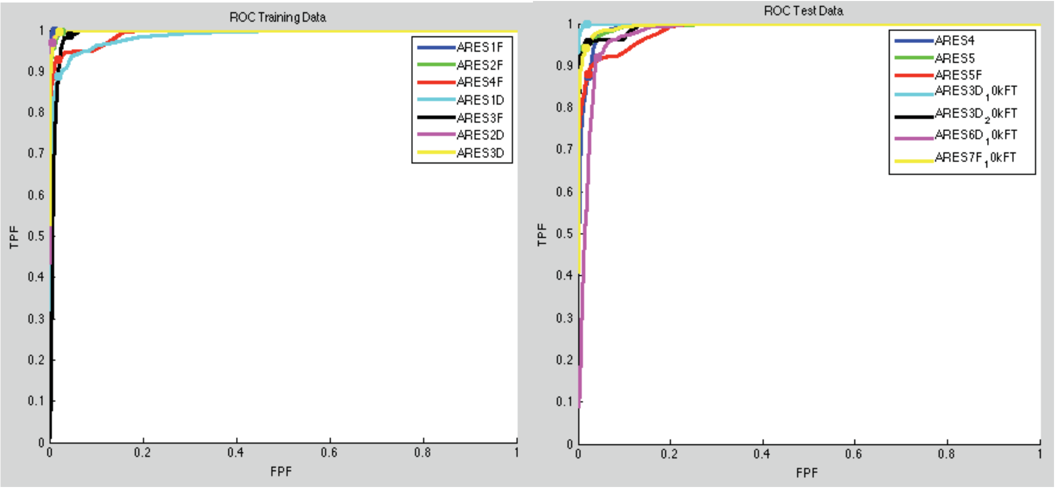


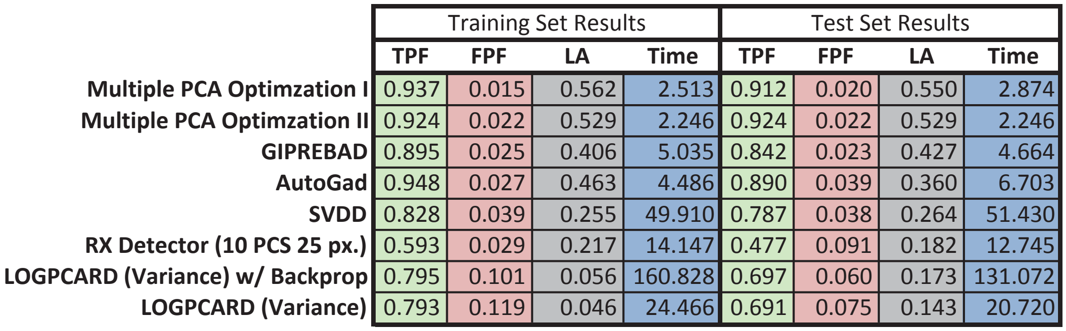
Figure 54: ROC Optimization II Results

IV. Results and Analysis

4.1 Chapter Overview

This chapter offers an in depth numerical analysis of the algorithms introduced in Chapter III. Average results comparing all algorithms considered in this research are shown in Table 13. The rows are sorted by LA on the test set. Contrasts between the Multiple PCA algorithm and the improved AutoGAD algorithm are of particular interest as they are the top performing algorithms. The GIPREBAD algorithm is also competitive, and so it will be compared to the AutoGAD algorithm as well. The logistic PCA reconstruction error based anomaly detector did not perform well past the two images used for algorithm development, but results will still be presented as future research may enable successful anomaly detection using deep belief nets, Boltzmann machines, and autoencoders.

Table 13:Average Results for Multiple Detectors Sorted by Test LA



	Training Set Results				Test Set Results			
	TPF	**FPF**	**LA**	**Time**	**TPF**	**FPF**	**LA**	**Time**
Multiple PCA Optimzation I	0.937	0.015	0.562	2.513	0.912	0.020	0.550	2.874
Multiple PCA Optimzation II	0.924	0.022	0.529	2.246	0.924	0.022	0.529	2.246
GIPREBAD	0.895	0.025	0.406	5.035	0.842	0.023	0.427	4.664
AutoGad	0.948	0.027	0.463	4.486	0.890	0.039	0.360	6.703
SVDD	0.828	0.039	0.255	49.910	0.787	0.038	0.264	51.430
RX Detector (10 PCS 25 px.)	0.593	0.029	0.217	14.147	0.477	0.091	0.182	12.745
LOGPCARD (Variance) w/ Backprop	0.795	0.101	0.056	160.828	0.697	0.060	0.173	131.072
LOGPCARD (Variance)	0.793	0.119	0.046	24.466	0.691	0.075	0.143	20.720

4.2 Multiple PCA and AutoGAD

The Multiple PCA algorithm showed statistically significant improvements over the AutoGAD algorithm in the important performance measure of label accuracy for the test set. In addition to this, the mean LA and TPF for Multiple PCA exceed that of AutoGAD for both training and test sets with both optimization I and optimization II settings. Furthermore, the mean FPF is less than that of AutoGAD for training and tests sets in both configurations as well. The bin size (sensitivity) parameter $Y = 300$ was selected for AutoGAD, and the optimal settings found by Johnson (2008), were used for all other settings in the algorithm. It is important to note that an RPD or RSM model was not created to re-optimize the AutoGAD parameters.

One key advantage of the Multiple PCA algorithm over AutoGAD is its deterministic nature. As described in 2.4.3, AutoGAD relies on an algorithm called FastICA that randomly generates initial solutions in its search for the independent components. Thus, even on the same image, the algorithms performance and time for completion vary. Generally performance characteristics were fairly consisted, but there were quite large variations in time required for algorithm completion. Because of this random nature, AutoGAD results shown will include an * in order to indicate they are the mean of 30 repetitions. This can be seen in Table 14 showing results for the training set. Here, the standard deviations are included to show the variations in label accuracy and time required for algorithm completion. The variance of the TPF and FPF values were very small and were not considered meaningful for the training or test sets used in this research and are thus not shown in the results. LA and TPF did vary enough to warrant consideration and as such their standard deviations are included.

Table 14: AutoGAD and Multiple PCA Training Results

	AUTOGAD						Multiple PCA				Multiple PCA			
	30 Reps, Y=300						Optimization II				Optimization I			
	TPF*	**FPF***	**LA***	**LA SD**	**Time***	**Time SD**	**TPF**	**FPF**	**LA**	**Time**	**TPF**	**FPF**	**LA**	**Time**
ARES1F	0.987	0.008	0.854	0.000	0.837	0.050	0.998	0.012	0.830	2.425	0.998	0.013	0.818	2.391
ARES2F	0.983	0.058	0.188	0.011	19.977	0.986	0.996	0.027	0.463	3.150	0.998	0.019	0.592	3.615
ARES3F	0.889	0.050	0.134	0.024	1.146	0.226	0.930	0.019	0.359	2.140	0.874	0.020	0.418	2.304
ARES4F	0.864	0.012	0.495	0.013	0.552	0.046	0.887	0.019	0.461	1.280	0.973	0.048	0.106	1.252
ARES1D	0.969	0.032	0.157	0.055	7.223	4.023	0.988	0.043	0.118	3.860	0.936	0.018	0.365	4.467
ARES2D	0.977	0.003	0.940	0.021	0.993	0.196	0.970	0.005	0.933	1.640	0.948	0.000	0.995	1.672
ARES3D	0.969	0.024	0.472	0.000	0.674	0.077	0.995	0.022	0.516	1.840	0.944	0.019	0.514	1.893
Mean	0.948	0.027	0.463	**0.018**	4.486	**0.801**	0.966	0.021	0.526	2.334	0.953	0.020	0.544	2.513

In general, the completion time for AutoGAD is quite quick as one can note in both Table 14 and Table 15 showing the test results. The algorithm runs in less than 1.2 seconds for 9 out of the 14 images used in this research, but for the remaining 5 images the completion time is far slower. Along with the difference in mean completion time is a drastic increase in time standard deviation. Referencing image sizes in Table 6 and Table 7 reveals that larger images, and those with higher Fisher scores, seem to cause longer processing times. Despite its general quickness, these variability issues might be problematic if attempting to integrate AutoGAD on a sensor platform with limited processing power due to this high level of variability in processing time. On the other hand, Multiple PCA is much more consistent and although Multiple PCA was only faster on 5 out of the 14 images, the mean processing time was still less than that of AutoGAD for both training and test sets. The processing times for multiple PCA are much more predictable and seem to increase linearly with image size which may represent a major advantage if considering sensor integration.

Table 15: AutoGAD and Multiple PCA Test Results

	AUTOGAD						Multiple PCA				Multiple PCA			
	30 Reps, Y=300						Optimization II				Optimization I			
	TPF*	FPF*	LA	LA SD	Time*	Time SD	TPF	FPF		Time	TPF	FPF		Time
ARES4	0.885	0.024	0.547	0.015	0.807	0.007	0.873	0.022	0.626	2.350	0.954	0.023	0.639	3.335
ARES5	0.900	0.039	0.250	0.003	1.040	0.009	0.941	0.014	0.604	3.670	0.936	0.009	0.706	5.034
ARES5F	0.916	0.066	0.260	0.024	31.046	6.314	0.879	0.021	0.561	4.870	0.879	0.023	0.557	6.815
3D_10kFT	0.930	0.043	0.284	0.020	0.743	0.037	1.000	0.020	0.523	1.060	1.000	0.024	0.488	1.217
3D_20kFT	0.811	0.016	0.384	0.000	0.601	0.355	0.955	0.021	0.483	0.566	0.944	0.026	0.431	0.628
6D_10kFT	0.863	0.065	0.147	0.020	9.326	2.320	0.919	0.042	0.230	1.410	0.810	0.028	0.266	1.480
7F_10kFT	0.923	0.018	0.648	0.000	3.360	0.088	0.941	0.018	0.686	1.280	0.954	0.013	0.748	1.290
Mean	0.890	0.039	0.360	**0.012**	6.703	**1.304**	0.930	0.022	0.531	2.172	0.925	0.021	0.548	2.828

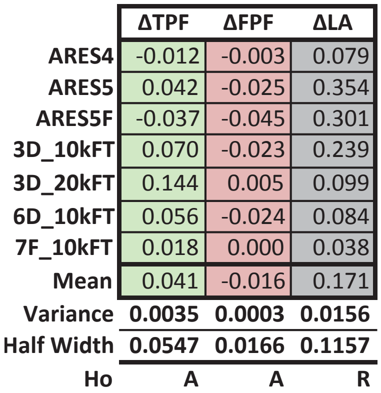
In comparing the algorithms through the test set results shown in Table 15, it is readily apparent that the Multiple PCA outperforms AutoGAD in all mean performance statistics. However, since all images are from the HYDICE sensor it could be that this is a matter of chance due to the slim margins separating the algorithms. For this reason, a paired *t*-test is employed to determine the statistical significance of performance characteristic differences. The paired test was chosen because of the correlation in results between the two algorithms from image to image; results on the same images for accurate anomaly detectors will most certainly not be independent. All tests were conducted at the .05 confidence level and formulated as:

$$\begin{aligned} H_0 &: P_{measureA} - P_{measureB} = 0 \\ H_A &: P_{measureA} - P_{measureB} \neq 0 \end{aligned} \tag{62}$$

Where $P_{measureA}$ and $P_{measureB}$ are the TPF, FPF, or LA for each algorithm. The null hypothesis is that the two performance measures are equal; while the alternate hypothesis is that they are not. In order to form the test, the corresponding results for each image were subtracted from each other to form the ΔTPF, ΔFPF, & ΔLA as shown in Table 16. The results for AutoGAD were subtracted from results for Multiple PCA; therefore, positive ΔTPF and ΔLA and negative ΔFPF indicate better performance for Multiple

PCA. It is important to note that the confidence intervals constructed assume it is appropriate to compare the statistics individually, and if they were compared simultaneously, the confidence intervals need to be considerably wider.

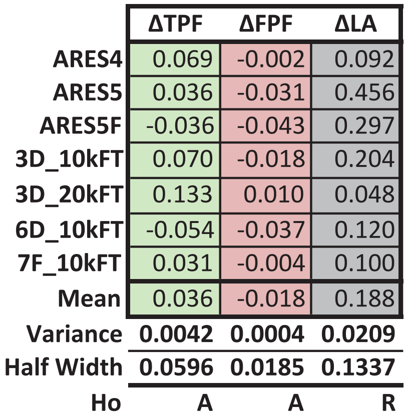
Table 16: Paired *t*-tests Multiple PCA Optimization II & AutoGAD



	ΔTPF	ΔFPF	ΔLA
ARES4	-0.012	-0.003	0.079
ARES5	0.042	-0.025	0.354
ARES5F	-0.037	-0.045	0.301
3D_10kFT	0.070	-0.023	0.239
3D_20kFT	0.144	0.005	0.099
6D_10kFT	0.056	-0.024	0.084
7F_10kFT	0.018	0.000	0.038
Mean	0.041	-0.016	0.171
Variance	0.0035	0.0003	0.0156
Half Width	0.0547	0.0166	0.1157
Ho	A	A	R

The results of three separate paired *t*-tests are shown in Table 16 comparing performance statistics for Multiple PCA optimization II and AutoGAD. The null hypothesis that label accuracies of both AutoGAD and Multiple PCA are equal is rejected. The corresponding p-values for the TPF, FPF, and LA tests are .124, .054, and .011. Therefore, Multiple PCA offers a statistically significant improvement in LA over AutoGAD. There is not enough evidence to conclude that the average TPF or FPF for the Multiple PCA configuration is greater than the TPF for AutoGAD. Similar results are shown in Table 17 for Multiple PCA Optimization I with corresponding p-values of .195, .056, and .014 for the TPF, FPF, and LA tests respectively. The significant improvement Multiple PCA offers in LA is very operationally meaningful. Higher label accuracy, also called positive predictive value, means that a higher percentage of anomalies declared are actually anomalies. Thus, utilizing results from Multiple PCA as opposed to AutoGAD would result in fewer resources wasted in the form of time spent by human analysts or computational time for spectral matching methods on false alarms.

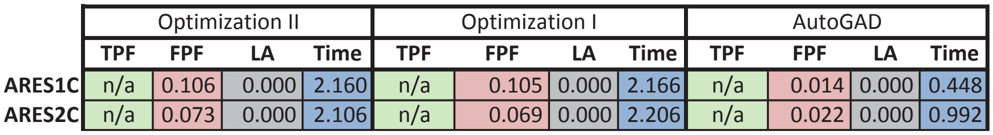
Table 17: Paired t-tests Multiple PCA Optimization I & AutoGAD



	ΔTPF	ΔFPF	ΔLA
ARES4	0.069	-0.002	0.092
ARES5	0.036	-0.031	0.456
ARES5F	-0.036	-0.043	0.297
3D_10kFT	0.070	-0.018	0.204
3D_20kFT	0.133	0.010	0.048
6D_10kFT	-0.054	-0.037	0.120
7F_10kFT	0.031	-0.004	0.100
Mean	0.036	-0.018	0.188
Variance	0.0042	0.0004	0.0209
Half Width	0.0596	0.0185	0.1337
Ho	A	A	R

Real world hyperspectral image patches of natural scenes will are often devoid of pixels of interest or targets. Conversely, the training and test sets were composed of staged hyperspectral images with a relatively high target density. For this reason it was imperative to validate the proposed anomaly detection methods for oversensitivity. Table 8 in section 3.2 described the validation set consisting of two images, ARES1C and ARES2C. These images do not contain any targets and can be used to determine whether proposed algorithms have unacceptably high false alarm rates in low target density regions. Unfortunately, the Multiple PCA algorithm seemed to have a troubling bias towards declaring targets in the two scenes. The results on the validation images for both AutoGAD and Multiple PCA are shown in Table 18.

Table 18: Initial Validation Results



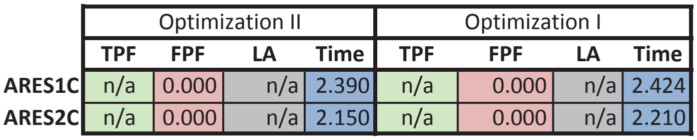
	Optimization II				Optimization I				AutoGAD			
	TPF	FPF	LA	Time	TPF	FPF	LA	Time	TPF	FPF	LA	Time
ARES1C	n/a	0.106	0.000	2.160	n/a	0.105	0.000	2.166	n/a	0.014	0.000	0.448
ARES2C	n/a	0.073	0.000	2.106	n/a	0.069	0.000	2.206	n/a	0.022	0.000	0.992

The results on the validation images may have rendered the Multiple PCA algorithm unusable. It was especially troubling considering that its most meaningful improvement over AutoGAD was its significant edge in Label Accuracy, while the validation set results for multiple PCA revealed a label accuracy of zero. The validation results certainly seem to confirm that AutoGAD is a more useful technique when considering its more reasonable FPFs. Thus, the algorithm was modified slightly from the code described in Figure 44 in order to address the high FPF rate. The fix came in the form of a test for detection score saliency using the signal to noise ratio of potential targets to background for each of the four D scores. Before the anomaly declaration votes of a particular score would be counted, its signal to noise needed to exceed a given threshold, SNR_{thresh}. As in AutoGAD (Johnson, Williams, & Bauer, 2013) the pixel variability of the background was considered a measure of the power of the noise and the variability of potential anomalous pixels was considered the power of the signal. The measure of signal to noise ratio is thusly calculated

$$SNR_{D_i} = 10\log_{10}\left(\frac{\mathrm{var}(\mathit{potential\_target\_signal}_{D_i})}{\mathrm{var}(\mathit{background}_{D_i})}\right) \tag{63}$$

Here, the potential target signal is the set of pixels declared as anomalous for each score on the algorithm's second iteration while the background is simply those pixels not declared anomalous. Some brief experimentation revealed that a threshold of $SNR_{thresh} = 7$ performed well. The results are reported in Table 19 where zero anomalies were declared in both of the validation images.

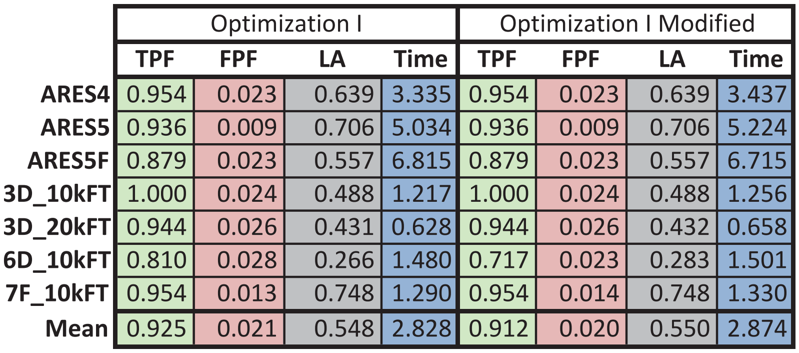
Table 19: Final Validation Results



	Optimization II				Optimization I			
	TPF	FPF	LA	Time	TPF	FPF	LA	Time
ARES1C	n/a	0.000	n/a	2.390	n/a	0.000	n/a	2.424
ARES2C	n/a	0.000	n/a	2.150	n/a	0.000	n/a	2.210

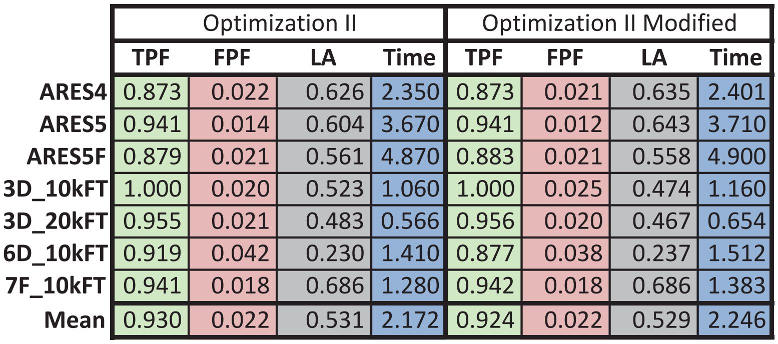
As would be expected, the signal to noise ratio based voting restriction also affected the operating characteristics of multiple PCA on both the training and test set. Fortunately, the results were not markedly different after this modification. The test results for the altered algorithm are shown alongside the original results in Table 20 and Table 21 for Optimization I and Optimization II, respectively. Note the slight increase in mean label accuracy for Optimization I with an accompanying decrease in the mean TPF and FPF rates. A similar decrease in mean TPF and FPF was noticed for the settings derived in Optimization II, but with an accompanying slight decrease in mean LA. Interestingly, the average performance deficits are largely the result of poor performance on just one image, 6D_10kFT.

Table 20: SNR Modification Results Optimization I



	Optimization I				Optimization I Modified			
	TPF	FPF	LA	Time	TPF	FPF	LA	Time
ARES4	0.954	0.023	0.639	3.335	0.954	0.023	0.639	3.437
ARES5	0.936	0.009	0.706	5.034	0.936	0.009	0.706	5.224
ARES5F	0.879	0.023	0.557	6.815	0.879	0.023	0.557	6.715
3D_10kFT	1.000	0.024	0.488	1.217	1.000	0.024	0.488	1.256
3D_20kFT	0.944	0.026	0.431	0.628	0.944	0.026	0.432	0.658
6D_10kFT	0.810	0.028	0.266	1.480	0.717	0.023	0.283	1.501
7F_10kFT	0.954	0.013	0.748	1.290	0.954	0.014	0.748	1.330
Mean	0.925	0.021	0.548	2.828	0.912	0.020	0.550	2.874

Table 21: SNR Modification Results Optimization II



	Optimization II				Optimization II Modified			
	TPF	FPF	LA	Time	TPF	FPF	LA	Time
ARES4	0.873	0.022	0.626	2.350	0.873	0.021	0.635	2.401
ARES5	0.941	0.014	0.604	3.670	0.941	0.012	0.643	3.710
ARES5F	0.879	0.021	0.561	4.870	0.883	0.021	0.558	4.900
3D_10kFT	1.000	0.020	0.523	1.060	1.000	0.025	0.474	1.160
3D_20kFT	0.955	0.021	0.483	0.566	0.956	0.020	0.467	0.654
6D_10kFT	0.919	0.042	0.230	1.410	0.877	0.038	0.237	1.512
7F_10kFT	0.941	0.018	0.686	1.280	0.942	0.018	0.686	1.383
Mean	0.930	0.022	0.531	2.172	0.924	0.022	0.529	2.246

Changes do not alter the results of the hypothesis tests performed above comparing AutoGAD with Multiple PCA. This is shown graphically in Figure 55 and Figure 56 for Optimizations I and II, respectively. The small blue dots show the actual deviations in the performance measures for the two algorithms, while the diamond represents the mean difference, a red line is drawn at zero, and 95% confidence intervals are shown in green. The corresponding p-values for the TPF, FPF, and LA tests are .169, .061, and .015 for Optimization I and .538, .057, and .013 for Optimization II. Once again, it is apparent that Multiple PCA significantly outperforms AutoGAD in LA while there is not a significant difference in performance in TPF and FPF. Once again, LA is a highly important performance measure because high label accuracies result in fewer resources expended on false alarms.

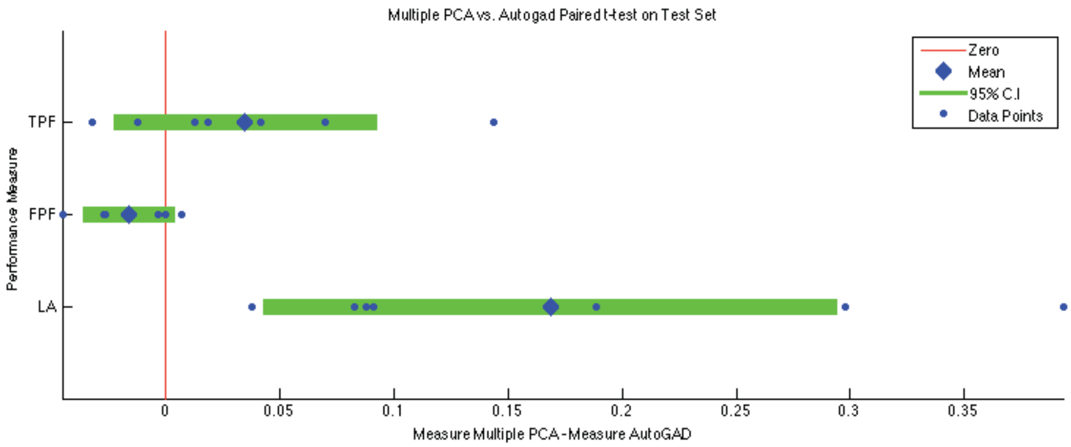


Figure 55: Optimization I Multiple PCA vs. AutoGAD (modified)

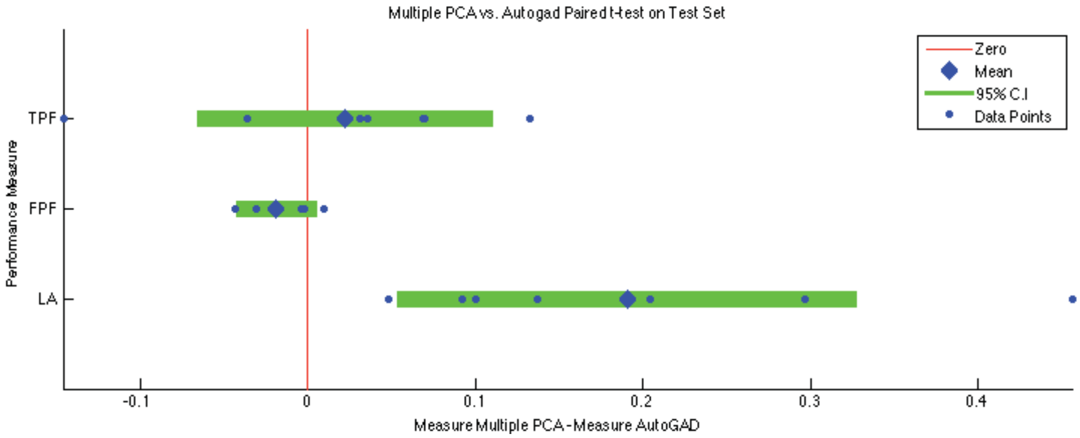
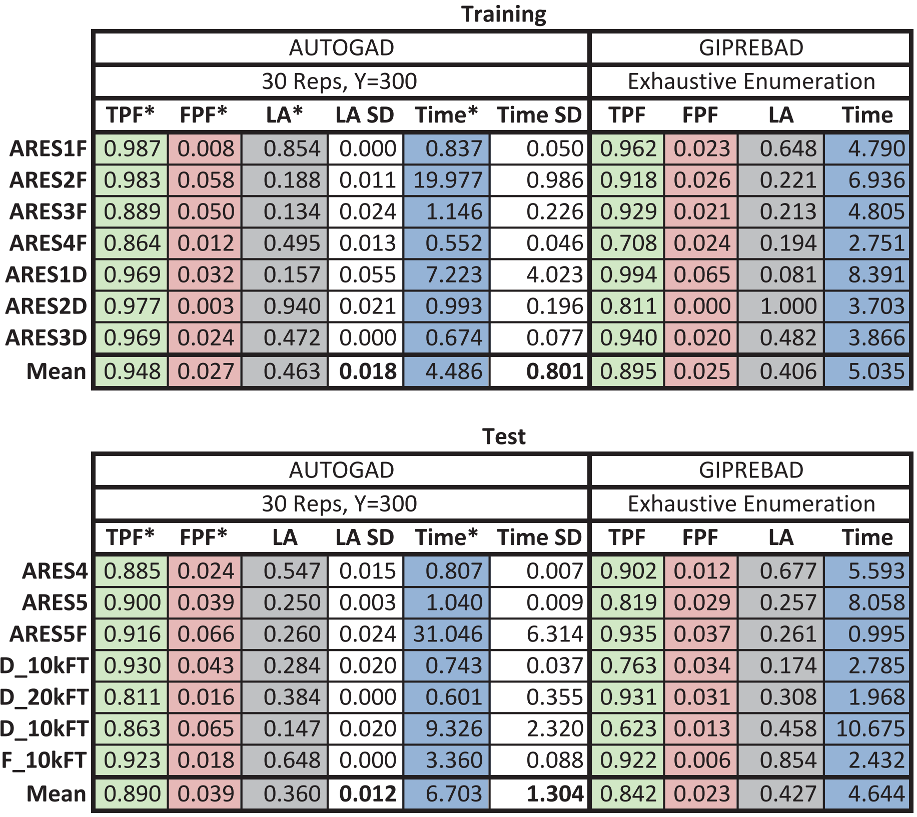


Figure 56: Optimization II Multiple PCA vs. AutoGAD (modified)

Perhaps unsurprisingly, there is not a statistically significant difference in the results from either Optimization I or Optimization II. Some slight operational differences do exist. The second optimization did offer slightly less processing times than the first. The time difference is generally due processing times required to construct histograms for the initial anomaly detection , as the $Y_{initial}$ setting for Optimization II is .249 as opposed to .142 for Optimization I. This results a very large number of bins for the first method and requires a large amount of memory. Optimization II does offer slightly better TPF rates, but at the cost of higher FPF and lower LA. Optimization II performance measures also were more sensitive to the signal to noise ratio modification. Further, it is more practical to adjust the single sensitivity parameter used in optimization I for final anomaly detection as opposed to four separate parameters in Optimization II; this might be a useful feature for an analyst. Thus, despite the time difference between the two optimizations, the author recommends Optimization I for selecting parameters on new image classes.

4.3 GIPREBAD and AutoGAD

Table 22: AutoGAD and GIPREBAD Results



Training

	AUTOGAD						GIPREBAD			
	30 Reps, Y=300						Exhaustive Enumeration			
	TPF*	**FPF***	**LA***	**LA SD**	**Time***	**Time SD**	**TPF**	**FPF**	**LA**	**Time**
ARES1F	0.987	0.008	0.854	0.000	0.837	0.050	0.962	0.023	0.648	4.790
ARES2F	0.983	0.058	0.188	0.011	19.977	0.986	0.918	0.026	0.221	6.936
ARES3F	0.889	0.050	0.134	0.024	1.146	0.226	0.929	0.021	0.213	4.805
ARES4F	0.864	0.012	0.495	0.013	0.552	0.046	0.708	0.024	0.194	2.751
ARES1D	0.969	0.032	0.157	0.055	7.223	4.023	0.994	0.065	0.081	8.391
ARES2D	0.977	0.003	0.940	0.021	0.993	0.196	0.811	0.000	1.000	3.703
ARES3D	0.969	0.024	0.472	0.000	0.674	0.077	0.940	0.020	0.482	3.866
Mean	0.948	0.027	0.463	**0.018**	4.486	**0.801**	0.895	0.025	0.406	5.035

Test

	AUTOGAD						GIPREBAD			
	30 Reps, Y=300						Exhaustive Enumeration			
	TPF*	**FPF***	**LA**	**LA SD**	**Time***	**Time SD**	**TPF**	**FPF**	**LA**	**Time**
ARES4	0.885	0.024	0.547	0.015	0.807	0.007	0.902	0.012	0.677	5.593
ARES5	0.900	0.039	0.250	0.003	1.040	0.009	0.819	0.029	0.257	8.058
ARES5F	0.916	0.066	0.260	0.024	31.046	6.314	0.935	0.037	0.261	0.995
3D_10kFT	0.930	0.043	0.284	0.020	0.743	0.037	0.763	0.034	0.174	2.785
3D_20kFT	0.811	0.016	0.384	0.000	0.601	0.355	0.931	0.031	0.308	1.968
6D_10kFT	0.863	0.065	0.147	0.020	9.326	2.320	0.623	0.013	0.458	10.675
7F_10kFT	0.923	0.018	0.648	0.000	3.360	0.088	0.922	0.006	0.854	2.432
Mean	0.890	0.039	0.360	**0.012**	6.703	**1.304**	0.842	0.023	0.427	4.644

The GIPREBAD algorithm performed fairly well despite it only actually representing one of the four members of the Multiple PCA ensemble. GIPREBAD performed better than AutoGAD on FPF and LA while falling quite a bit behind in TPF as shown in Table 22. The confidence intervals in Figure 57 show no statistical difference between the two algorithms. The p-values for the TPF, FPF, and LA tests are .165, .093, and .294, respectively. Unfortunately, the assumption of correlation between the detectors in response to given imagery does come into question here, and as a result the paired *t*-test results may be misleading. This is especially true in the TPF response where there is a large mean difference, but the extreme variability of the score differences causes the confidence interval to be quite large.   The author speculates that the difference would be statistically significant with a larger sample size.

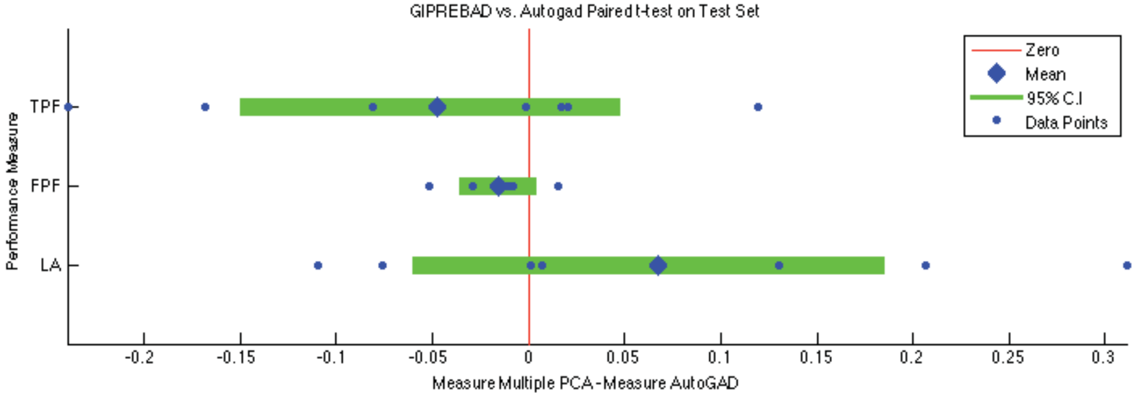
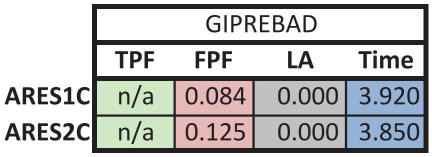


Figure 57: Paired *t*-tests AutoGAD & GIPREBAD

The GIPREBAD algorithm is also quite a bit slower than Multiple PCA and timing is highly variable. GIPREBAD thus suffers from the same inconsistency in algorithm completion time that AutoGAD does and also would be difficult to implement on a sensor platform with limited computational resources. The iterative nature of GIPREBAD increases the time variability, and perhaps limiting the algorithm to one iteration would alleviate this problem but perhaps with a decrease in performance.

Table 23: GIPREBAD Validation Results



	GIPREBAD			
	TPF	FPF	LA	Time
ARES1C	n/a	0.084	0.000	3.920
ARES2C	n/a	0.125	0.000	3.850

GIPREBAD also performed very poorly on the validation images as evidenced in Table 23. Unfortunately, a signal to noise ratio threshold in this algorithm is not quite as practical as in Multiple PCA due to a single score being used to elect anomalies. The high false alarm rate evident in these validation images most likely renders the algorithm useless in real-world applications. The algorithm results are, however, fairly promising and methods of alleviating this problem should be explored in future research.

4.4 Logistic PCA Reconstruction Error Anomaly Detection Results

The LogPCARD algorithm exhibited the poorest performance out of all the methods presented. First and foremost, it was slow. Anomaly detection using only contrastive divergence to train the individual restricted Boltzmann machines took 17.0 seconds on average. Adding ‘fine-tuning’ or backpropagation on the entire autoencoder required an average of 109.8 seconds to complete. Interestingly, backpropagation did not result in meaningfully better performance characteristics as evidenced by the results presented in Table 24. This is in contrast to the results for the two images used in algorithm development in Chapter III where ‘fine-tuning’ seemed to increase anomaly detection performance.

Table 24: Logistic PCA Combined Training & Test Results

	Mean Values						Variances					
	No Backprop			w/ Backprop			No Backprop			w/ Backprop		
	TPF*	FPF*	LA*	TPF*	FPF*	LA*	TPF*	FPF*	LA*	TPF*	FPF*	LA*
Pixel Sum Squared Recon. Error	0.750	0.086	0.164	0.758	0.098	0.165	0.022	0.001	0.027	0.024	0.004	0.033
Pixel Recon. Error Variance	0.774	0.087	0.153	0.786	0.088	0.168	0.025	0.002	0.018	0.022	0.002	0.022
Variance Not Explained	0.726	0.076	0.236	0.739	0.084	0.235	0.055	0.003	0.043	0.054	0.004	0.047

Thus, adding backpropagation is likely not a practical choice for anomaly declaration in HSI with the current algorithm configuration. Of course, reducing the number of 'fine-tuning' epochs performed could prove to affect more reasonable processing times while still maintaining some improvements. Further, a small amount of experimentation indicated that overtraining could be reducing the signal to noise ratio of the anomalies to the background and thus a smaller number of epochs for some images might enable better results. Automating the monitoring of overtraining during backpropagation proved difficult and thus a consistent number of maximum iterations was chosen. Future research might explore better ways of finding an acceptable balance with respect to this issue.

It is important to note that the results in Table 24 were calculated using results from both the training and test sets to obtain more accurate performance estimations. The algorithm was not trained on the test set and therefore it made little sense to divide the data for analysis. Means and Variances in the performance metrics across all of the images for each of the proposed anomaly detection scores are shown in order to enable comparison between Q, σ_R^2, and R^{2*}. Here, we see that the pixel reconstruction error variance, σ_R^2, generally shows the best and most consistent performance as it shows generally favorable means with modest levels of variance. R^{2*} boasts the highest average LA, but all of its performance metrics maintain a very high variance. This is most likely due to its calculation requiring the comparison of two variances.

4.5 Conclusions

The Multiple PCA algorithm outperformed all of the methods considered in this research. The voting ensemble facilitated consistent and accurate anomaly declarations despite noise and varying image scenes. A SNR based voting threshold prevents anomaly declarations with sparsely targeted scenes. GIPREBAD offered acceptable performance levels but at the price of longer processing times and less consistency as well as horrible results on the validation images. LogPCARD shows some potential, but the algorithm would not be practical for anomaly detection in HSI in its current form.

V. Discussion

5.1 Limitations

All of the HYDICE data used for algorithm testing and optimization in this research were captured in rural scenes, while urban environments would likely present a much less homogenous background and would most certainly create problems for the PCA based detectors. The targets detected by the algorithms also only consisted of a few different vehicles, tarps, and tents. Actual 'real-world' applications would contain a much more diverse target set with possibly no true knowledge of objects within a given scene. Furthermore, the algorithms were developed across only a small set of forest and desert scenes collected under the Hyperspectral Digital Imagery Collection Experiment. HSI collected in different types of scenes, and with different sensors could require new algorithm parameters and further optimization.

Although the Multiple PCA algorithm detected targets accurately and effectively in the set of HYDICE data set used in this research, some limitations of the algorithm should be considered. One is the troubling high false positive rate on the two validation images, ARES1C and ARES2C before algorithm modification. The SNR based voting threshold set to compensate for this is highly specific to the two validation images. Further testing and evaluation would be required to confirm that this method would operate consistently across other scenes that are sparsely populated with targets. Furthermore, despite the algorithm being relatively quick, it still demands a large amount of processing power. This is due to repeated sorting and histogram construction and to a lesser extent, a large amount of IAN filtering. Despite this, average processing times are still faster than the AutoGAD algorithm while offering better performance.

5.2 Suggestions for Future Research

1) Re-optimize the AutoGAD algorithm with the new bin size parameter method through RPD or perhaps the RSM presented herein.
2) Explore different structures and methods for the LogPCARD algorithm. Gaussian inputs may represent the data better and be feasible with the right training parameters and enough processing power (attempt to use multiple graphics processor based training).
3) Explore utilizing the Multiple PCA algorithm for other high dimensional data anomaly detection problems such as network intrusion or credit fraud prevention.
4) Explore improving the GIPREBAD algorithm in order to limit false positives in sparsely targeted scenes through SNR or other methods.
5) Customize sorting and histogram construction methods in Multiple PCA to avoid redundancy and lessen processing requirements.

5.3 Original Contributions to the Field of Anomaly Detection

1) PCA reconstruction error methods applied to HSI anomaly detection.
2) Iterative anomaly detection using PCA reconstruction error such as to better approximate the background distribution.
3) The voting ensemble of the Multiple PCA algorithm.
4) Use of the median whitened PC score as a robust estimator to the Mahalanobis distance for anomaly detection.
5) Logistic Principal Component Reconstruction Error Based Hyperspectral Anomaly Detection
6) Introduced the “variance not explained” and the variance of reconstructive error scores for use in anomaly detection.
7) Improvements on AutoGAD enabling ROC analysis and the automatically adjustable bin-size parameter.

5.4 Conclusions

On October 22, 2010 The Defense Advanced Research Projects Agency (DARPA) announced a project entitled “Anomaly Detection at Multiple Scales” (ADAMS). Thirty-five million dollars in funding was allocated for the two-year research mission intended to develop methods to detect and prevent insider threats such as the 2009 Fort Hood shooting. In order to succeed, weak signals of anomalous behavior would need to be detected within a noisy background of normal behavior. Part of the announcement reads:

> “The general goal of the ADAMS program is to create, adapt and apply technology to the problem of anomaly characterization and detection in massive data sets. The importance of anomaly detection is due to the fact that anomalies in data translate to significant, and often critical, actionable information in a wide variety of application domains…. While technology developed for ADAMS will have applicability in many domains, we will use the problem of insider threat detection as a focal point in order to make sure that the work is well grounded (Defense Advanced Research Projects Agency, 2010).”

Although the ADAMS program focused on a much different problem, the relevance of anomaly detection is undeniable. Data abundance seems to be eclipsing the speed of processors and thus efficient and effective algorithms that are able to handle large amounts of data are required. The multiple PCA algorithm offers just such a method as it is simple, efficient, and accurate and also likely would be effective in other application domains where high dimensionality and noise increase detection difficulties and inflate false alarm rates.

Bibliography

Banerjee, A., Burlina, P., & Diehl, C. (2006). A Support Vector Method for Anomaly Detection in Hyperspectral Imagery. *IEEE Transactions on Geoscience and Remote Sensing , 44* (8), 2282-2291.

Becker, D., King, T. D., McMullen, B., & Fahsi, A. (2013). Big Data Quality Case Study Preliminary Findings: Hyperspectral Imaging (HSI) Using the Airborne Visible/INfrared Imaging Spectrometer (AVIRIS). *The 18th International Conference on Information Quality.* Little Rock: University of Arkansas.

Bedikstsson, J. A., Palmason, J. A., & Sveinsson, J. R. (2005, Mar.). Classification of hyperspectral data from urban areas based on extended morphological profiles. *IEEE Transactions on Geoscience and Remote Sensing* , 480-491.

Bengio, Y., Delalleau, O., Le Roux, N., Vincent, Vincent, P., & Oimet, M. (2004). *Spectral Dimensionality Reduction.* Universit´e de Montr´eal , Centre de Recherches Math´ematiques.

Bigley, A. L. (2013). *Horn's Curve Estimation Through Multi-Dimensional Interpolation.* WPFAB: Air Force Institute of Technology.

Brand, J. C. (1995). *Lines of Light.* Amsterdam: Overseas Publishers Association.

Bush, K. R. (2012). *Using QR Factorization For Real-time Anomaly Detection of Hyperspectral Images.* WPAFB: Air Force Institute of Technology.

Chan, T. H., Ni, Y. Q., & Ko, J. M. (1999). Neural network novelty filtering for anomaly detection of Tsing Ma Bridge cables. *Proceedings of the 2nd International Workshop on Structural Health Monitoring,* (pp. 430-439).

Chandola, V., Banerjee, A., & Kumar, V. (2007). *Anomaly Detection: A Survey* . Department of Computer Science and Engineering . Minneapolis: University of Minnesota.

Chen, C. H., & Zhang, X. (1999). Independent component analysis for remote sensing study. *EOS/SPIE Symposium on Remote sensing. 3871*, pp. 150-158. Florence, Italy: SPIE.

Cherivadat, A., & Bruce, L. M. (2003). Why Principal Component Analysis is not an Appropriate Feature Extraction Method for Hyperspectral Data. *Geoscience and Remote Sensing Symposium. 6*, pp. 3420-3422. IGARSS.

Chiang, S.-S., Chang, C.-I., & Ginsber, I. W. (2001). Unsupervised Target Detection in Hyperspectral Images Using Projection Pursuit. *IEEE Transactions on Geoscience and Remote Sensing , 29* (7), 1380-1397.

Chiang, S.-S., Chang, C.-l., & Ginsber, I. W. (2000). Unsupervised Hyperspectral Image Analysis Using Independent Component Analysis. *Geoscience and Remote Sensing Symposium . 7*, pp. 3136-3138. Honolulu: IGARSS.

Christophe, E. (2011). Hyperspectral Data Compression Tradeoff. In S. Prasad, L. M. Bruce, & J. Chanussot, *Optical Remote Sensing, Advances in Signal Processing and Exploitation Techniques* (Vol. 3, pp. 9-29). Berlin: Springer.

Defense Advanced Research Projects Agency. (2010, October 22). *Anomaly Detection at Multiple Scales (ADAMS), Broad Agency Announcement.* Retrieved February 5, 2014, from Federal Business Opportunities: https://www.fbo.gov/download/2f6/2f6289e99a0c04942bbd89ccf242fb4c/DARPA-BAA-11-04_ADAMS.pdf&ei=6wQYU_eMJ-_iyAHN4IHYCw&usg=AFQjCNFqOSJ7VNoZ7j-oXg98SAIufv-MJA&bvm=bv.62577051,d.aWc

Dillon, W. R., & Goldstein, M. (1984). *Multivariate Analysis.* New York: John Wiley & Sons.

Ding, Q., & Kolacyk, E. D. (2010, August 3). A Compressed PCA Subspace Method for Anomaly Detection in High-Dimensional Data . *Joint Statistical Meeting*.

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification.* New York: John Wiley & Sons.

Eismann, M. T. (2012). *Hyperspectral Remote Sensing.* Bellingham, WA: SPIE.

Elderding, G. T., Thunen, J. G., & Woody, L. M. (1991). Wedge imaging spectrometer: application to drug and pollution law enforcement. *Surveillance Technologies.* Orlando, FL: SPIE.

Farrell, M. D., & Mersereau, R. M. (2005). On the Impact of PCA Dimensionality Reduction for Hyperspectral Detection of Difficult Targets. *IEE Geoscience and Remote Sensing Letters , 2* (2), 192-195.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters , 27*, 861-874.

Flamm, D. (1983). Ludwig Boltzmann and his Influence on Science. *Studies in History and Philosophy of Science Part A , 14* (4), 255-278.

Fountanas, L. (2004). *Principlal Components Based Techniques for Hyperspectral Image Data.* MS Thesis, Naval Postgraduate School, Monterey, CA.

Goldstein, S. (2002). Boltzmann's Approach to Statistical Mechanics. *Chance in Physics, Foundations*, (p. 39).

Hinton, G. E. (2010). *A Practical Guide to Training Restricted Boltzmann Machines.* Technical, University of Toronto, Department of Computer Science, Toronto.

Hinton, G. E. (2007, July). Reducing the Dimensionality of Data with Neural Networks. *SCIENCE* , 504-507.

Hinton, G. E. (2007). The Next Generation of Neural Networks. *Google Tech Talks*.

Hinton, G. E., & Sejnowski, T. J. (1984). *Boltzmann Machines: Constraint Satisfaction Networks That Learn.* Tech. Rep., Carnegie-Mellon University, Computer Science, Pittsburg, PA.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *J. Educ. Psychol. , 24*, 417-441.

Hyvärinen, A. (1999). Fast and Robust Fixed-Point Algorithms for Independent Component Analysis. *IEEE Transactions on Neural Networks , 10* (3), 626-634.

Jackson, D. A. (1993). Stopping rules in principal components analysis: a comparison of heuristical and statistical approaches. *Ecology* , 2204-2214.

Jackson, E. J., & Morris, R. H. (1957). An Application of Multivariate Quality Control to Photographic Processing . *Journal of the American Statistical Association , 52* (278), 186-199.

Jackson, E. J., & Mudholkar, G. S. (1979). Control Procedures for Residuals Associated with Principal Component Analysis . *Technometrics , 21* (3), 341-349.

Johnson, R. J. (2008). *Improved Feature Extraction, Feature Selection, and Identification Techniques that Create a Fast Unsupervised Hyperspectral Target Detection Algorithm.* Thesis, Air Force Institute of Technology, WPAFB.

Johnson, R. J., Williams, J. P., & Bauer, K. W. (2013). AutoGAD: An Improved ICA-Based Hyperspectral Anomaly Detection Algorithm. *IEEE Transactions on Geoscience and Remote Sensing , 51* (6), 3492 - 3503.

Jolliffe, I. T. (2002). *Principle Component Analysis.* New York: Springer.

Kaski, S. (1998). Dimensionality Reduction by Random Mapping: Fast Similarity Computation for Clustering. *IEEE Internation Joint Conference on Neural Networks* .

Kindermann, R., & Snell, L. (1950). Markov Random Fields and Their Applications. *Comtemporary Mathematics* , 24-25.

Kittel, C., & Kroemer, H. (1980). *Thermal Physics.* New York: W.H. Freeman and Company.

Kramer, A. (1991). Nonlinear Principal Component Analysis Using Autoassociative Neural Networks. *AlChE , 37* (2), 233-243.

Krizhevsky, A. (2009). *Learning Multiple Layers of Features from Tiny Images.* Toronto: University of Toronto.

Landgrebe, D. (2002). Hyperspectral Image Data Analysis as a High Dimensional Signal Processing Problem . *IEEE Signal Processing Magazine , 19* (1), 17-28.

Licciardi, G., Del Frate, F., Schiavon, G., & Solimini, D. (2010). Dimensionality Reduction of Hyperspectral Data: Assessing the Performance of Autoassociative Neural Networks. *IGARSS* , 4377-4380.

Lin, D., & Tu, W. (1995). Dual Response Surface Optimization. *Journal of Quality Technology , 27*, 34-39.

Manolakis, D. (2002). *Detection Algorithms for Hyperspectral Imaging Applications.* Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, MA.

Manolakis, D. (2002). *Detection Algorithms for Hyperspectral Imaging Applications.* Massachusetts Institute Of Technology, Lincoln Laboratory , Lexington, MA.

Mindrup, F. M. (2011). *Optimized Hyperspectral Imagery Anomaly Detection through Robust Parameter Design.* WPAFB: Air Force Institute of Technology.

Montgomery, D. C. (2009). *Design and Analysis of Experiments* (7th Edition ed.). Wiley.

Myers, R. H., Montgomery, D. C., & Anderson-Cook, C. M. (2009). *Response Surface Methodology: Process and Product Optimization Using Deisgned Experiments.* Hoboken, NJ: John Wiley & Sons.

Myers, R., & Montgomery, D. (2002). *Response Surface Methodology. Process and Product Optimization Using Designed Experiments* (2nd Edition ed.). Wiley.

Nasrabadi, N. M. (2014, January). Hyperspectral Target Detection. *IEEE Signal Processing Magazine* , 34-45.

Nasrabadi, N. M., & Kwon, H. (2005). Hyperspectral Target Detection Using Kernel Orthogonal Subspace Projection. *IEEE International Conference on Image Processing. 2*, pp. 702-705. Genoa: ICIP.

Nischan, M. L., Kerekes, J. P., & Baum, J. E. (1999). Analysis of HYDICE Noise Characteristics and Their Impact on Subpixel Object Detection. *SPIE Conference on Imaging Spectrometry* (pp. 112-123). Dever: SPIE.

Osborne, B., Fearn, T., & Hindle, P. (1993). *Practical NIR Spectroscopy with Applications in Food and Beverage Analysis.* Essex: Longman Scientific & Technical.

Peres-Neto, P. R., Jackson, D. A., & Somers, K. M. (2003). Giving meaningful interpretation to orindation axes: assessing loading significance in principal component analysis. *Ecology , 84* (9), 2347-2363.

Prasad, S., & Bruce, L. M. (2008, Oct.). Limitiations of Principal Components Analysis for Hyperspectral Target Recognition". *IEEE Geoscience and Remote Sensing Letters*, 625-629.

Ranzato, M. A., Krizhevsky, A., & Hinton, G. E. (2010). Factored 3-Way Restricted Boltzmann Machines for Modeling Natural Images. *13th International Conference on Artificial Intelligence and Statistics. 9.* Sardinia: JMLR.

Reed, I. S., & Yu, X. (1990). Adaptive multiband CFAR detection of an optical pattern with unknown spectral distribution . *IEEE Transactions on Acoustics, Speech and Signal Processing , 38*.

Robila, S. A., & Varshney, P. K. (2002). Target Detection in Hyperspectral Images Based on Independent Component Analysis . *12th SPIE Automatic Target Recognition. 4726*, pp. 173-182. SPIE.

Rodionova, O. Y. (2005). NIR Spectrometry for Counterfeit Drug Detection: A Feasibility Study. *Analytica Chimica Acta , 549* (1-2), 151-158.

Sandidge, J. C., & Holyer, R. J. (1998). Coastal Bathymetry from Hyperspectral Observatons of Water Radiance. *Remote Sensing of Environment , 65* (3), 341-352.

Shan, J., & Rodarmel, C. (2002). Principal Component Analysis for Hyperspectral Image Classification. *Surveying Land Information Systems , 62* (2), 115-123.

Smetek, T. E. (2007). *Hyperspectral Imagery Target Detection Using Improved Anomaly Detection and Signature Matching Methods.* Wpright Patterson AFB OH: Air Force Insitute of Techology.

Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. In D. E. Rummelhart, & J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations.*. Cambridge: MIT Press.

Stein, D. W., Beaven, S. G., Hoff, L. E., Winter, E. M., Schaum, A. P., & Stocker, A. D. (2002, January). Anomaly Detection from Hyperspectral Imagery. *IEEE Signal Processing Magazine* , pp. 58-69.

Stocker, A. D., Ensafi, E., & Oliphant, C. (2003). Applications of eigenvalue distribution theory to hyperspectral processing. *Proceedings of SPIE, 5093*, pp. 652-664.

Stone, J. V. (2004). *Independent Component Analysis.* Cambridge, MA: The MIT Press.

Taitano, Y. P., Geier, B. A., & Bauer, K. W. (2010). A Locally Adaptable Iterative RX Detector. *EURASIP Journal on Advances in Signal Processing*.

Trussell, G. S. (1997). Digital Color Imaging. *IEEE Transactions on Image Processing, 6* (7).

Williams, J. P., Bihl, T. J., & Bauer, K. W. (2013). Towards the mitigation of correlation effects in anomaly detection for hyperspectral imagery. *Journal of Defense Modeling and Simulation*.

Williams, J. (2012). *Towards the Mitigation of Correlation Effects in the Analysis of Hyperspectral Imagery with Extensions to Robust Parameter Design.* Air Force Institute of Technology, Department of Operational Sciences.

Wong, G. (2009). Anomaly Detection Rudiments for the Application of Hyperspectral Sensors in Aerospace Remote Sensing . *Journal of Physics: Conference Series 178, Sensors & their Applications XV* (pp. 1-7). IOP Publishing.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 074-0188*

1. REPORT DATE *(DD-MM-YYYY)* 03-27-2014	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From – To)* Sep 2012 – Mar 2014

TITLE AND SUBTITLE Reconstruction Error and Principal Component Based Anomaly Detection in Hyperspectral imagery	**5a. CONTRACT NUMBER** **5b. GRANT NUMBER** **5c. PROGRAM ELEMENT NUMBER**
6. AUTHOR(S) Jablonski, James A., Captain, USA	**5d. PROJECT NUMBER** **5e. TASK NUMBER** **5f. WORK UNIT NUMBER**
7. PERFORMING ORGANIZATION NAMES(S) AND ADDRESS(S) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/ENS) 2950 Hobson Way, Building 640 WPAFB OH 45433-8865	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT-ENS-14-M-11
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/RYA POC: Lt Col David Ryer, David.Ryer@us.af.mil 2241 Avionics Circle Area B, Building 620 WPAFB, OH 45433-7321 (937)528-8389	**10. SPONSOR/MONITOR'S ACRONYM(S)** **AFRL/RHIQ (example)** **11. SPONSOR/MONITOR'S REPORT NUMBER(S)**

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRUBTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES
This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

The rapid expansion of remote sensing and information collection capabilities demands methods to highlight interesting or anomalous patterns within an overabundance of data. This research addresses this issue for hyperspectral imagery (HSI). Two new reconstruction based HSI anomaly detectors are outlined: one using principal component analysis (PCA), and the other a form of non-linear PCA called logistic principal component analysis. Two very effective, yet relatively simple, modifications to the autonomous global anomaly detector are also presented, improving algorithm performance and enabling receiver operating characteristic analysis. A novel technique for HSI anomaly detection dubbed "multiple PCA" is introduced and found to perform as well or better than existing detectors on HYDICE data while using only linear deterministic methods. Finally, a response surface based optimization is performed on algorithm parameters such as to affect consistent desired algorithm performance.

15. SUBJECT TERMS
Principal Component Analysis, Anomaly Detection, Hyperspectral, Boltzmann Machines, Reconstruction Error, Logistic PCA, Response Surface Methodology

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dr. Kenneth W. Bauer, AFIT/ENS
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UU	141	**19b. TELEPHONE NUMBER** *(Include area code)* (937) 255-3636, ext 4328 (kenneth.bauer@afit.edu)