MESOSCALE, RADIOMETRICALLY REFERENCED, MULTI-TEMPORAL HYPERSPECTRAL DATA FOR CO_2 LEAK DETECTION BY LOCATING SPATIAL VARIATION OF BIOPHYSICALLY RELEVANT PARAMETERS

by

Cooper Patrick McCann

A dissertation submitted in partial fulfillment
of the requirements for the degree

of

Doctor of Philosophy

in

Physics

MONTANA STATE UNIVERSITY
Bozeman, Montana

April 2017

DEDICATION

To future students reading this thesis
You can do it, there is a light at the end of the tunnel, and it is not an oncoming dragon.

To Dr. Kevin Repasky
You took a chance with me and I'm glad we were able to work together on some cool science. It has been a wonderful experience.

To my wife
I could have done the academics without you, but I would not be the person that I've become without you doing it with me. I am better because of you.

To my mom
A tireless supporter of my education. Thank you for all your sacrifices over the years.

To my dad
Who has been there in the tough times and has always supported me.

To my children
May you always know that you can achieve what you set your heart on.

To Keith Johnson
For hours of distractions, dozens of board games, and too many super-cool physics discussions to count. It has been a long journey, but well worth it.

ACKNOWLEDGEMENTS

First and foremost, I need to thank my advisor, Dr. Kevin Repasky, for allowing me the opportunity to work on such an interesting project, for putting up with my sometime sporadic work schedule, and for the hundreds (thousands?) of pages of writing that he read/edited (and on more than one occasion total rewrite) for me. Also to the other collaborators on this project, Dr. Scott Powell, Dr. Rick Lawrence, and Mikinda Morin.

I must of course acknowledge my family, that while they may have been the cause of erratic work hours, were always supportive. Isabella & Elliott who more than once informed me that I needed to "leave and go to work." Tristen, who did the best job of greeting me every day when I came home. April who came into our family late in this journey of graduate school, but has made our family more complete. Finally, to my wife, who put up with physics stories/jokes and has been ever present in the hard stuff of life.

To all those who have given me the time and opportunity to "Physics" over the years, thank you. Caleb, Shane, Keith, Ana, and all my other classmates for helping me survive classes with newborns at home. Dr. Babbitt for helping me study for the comprehensive exam, for letting me pick my oral exam topic, and for being willing to be honest with me even if it was hard. The excellent professors in Physics and ECE who I had the honor to learn from. The awesome administrative staff: Margaret, Sarah, Hannah, Norm, Jeremy, Jerry, and all other others behind the scenes.

This work was supported by the U. S. Department of Energy and the National Energy Technology Laboratory under Award No. DE-FC26-05NT42587. Landsat data and Landsat Surface Reflectance product (LaSRC) courtesy of the U.S. Geological Survey.

TABLE OF CONTENTS

TABLE OF CONTENTS — CONTINUED

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

LIST OF FIGURES — CONTINUED

Figure Page

LIST OF FIGURES — CONTINUED

Figure Page

ABSTRACT

Low-cost flight-based hyperspectral imaging systems have the potential to provide valuable information for ecosystem and environmental studies as well as aide in land management and land health monitoring. This thesis describes (1) a bootstrap method of producing mesoscale, radiometrically-referenced hyperspectral data using the Landsat surface reflectance (LaSRC) data product as a reference target, (2) biophysically relevant basis functions to model the reflectance spectra, (3) an unsupervised classification technique based on natural histogram splitting of these biophysically relevant parameters, and (4) local and multi-temporal anomaly detection.

The bootstrap method extends standard processing techniques to remove uneven illumination conditions between flight passes, allowing the creation of radiometrically self-consistent data. Through selective spectral and spatial resampling, LaSRC data is used as a radiometric reference target. Advantages of the bootstrap method include the need for minimal site access, no ancillary instrumentation, and automated data processing. Data from a flight on 06/02/2016 is compared with concurrently collected ground based reflectance spectra as a means of validation achieving an average error of 2.74%.

Fitting reflectance spectra using basis functions, based on biophysically relevant spectral features, allows both noise and data reductions while shifting information from spectral bands to biophysical features. Histogram splitting is used to determine a clustering based on natural splittings of these fit parameters. The Indian Pines reference data enabled comparisons of the efficacy of this technique to established techniques. The splitting technique is shown to be an improvement over the ISODATA clustering technique with an overall accuracy of 34.3/19.0% before merging and 40.9/39.2% after merging. This improvement is also seen as an improvement of kappa before/after merging of 24.8/30.5 for the histogram splitting technique compared to 15.8/28.5 for ISODATA.

Three hyperspectral flights over the Kevin Dome area, covering 1843 ha, acquired 06/21/2014, 06/24/2015 and 06/26/2016 are examined with different methods of anomaly detection. Detection of anomalies within a single data set is examined to determine, on a local scale, areas that are significantly different from the surrounding area. Additionally, the detection and identification of persistent anomalies and non-persistent anomalies was investigated across multiple data sets.

INTRODUCTION

In the introduction, a brief overview of remote sensing will be presented followed by background associated with the problem being addressed, namely CO_2 leak detection. Vegetation stress will be introduced as a means of detecting a CO_2 leak since this stress can be detected using remote sensing techniques. A description of hyperspectral sensors and how they relate to being able to detect vegetation stress and why the specific sensor platform, small aircraft aerial imaging, was chosen. Lastly, the motivation for the work in this thesis will be discussed.

Remote Sensing Overview

Simply put, remote sensing is acquiring information about an object without being in physical contact with the object. This is done by sensing and recording reflected or emitted energy and then processing and analyzing that information (Elachi & Van Zyl, 2006; Nowatzki et al., 2004; Purkis & Klemas, n.d.; Santosh & Sundaresan, 2014). In this thesis, the focus will be on remote sensing of the earth from aircraft and satellites. Broadly, remote sensing has 7 elements:

1. An energy source or illumination provides electromagnetic energy to the target of interest. For many applications, this is accomplished by the sun, though the source could be manmade or other naturally occurring illumination.
2. Interaction with the atmosphere involves the incident radiation traversing the atmosphere on its way to the target, potentially being absorbed or scattered.

Additionally, the reflected/emitted radiation also will have to traverse the atmosphere after interacting with the target and will be subject to the same effects.

3. Interaction with the target will depend on what type of radiation and target are involved, and thus will define the absorption and reflection of the incident radiation.
4. Recording the reflected/emitted radiation by a sensor occurs remotely (not in direct contact). Again, the type of sensor will depend on several considerations, primarily the wavelength of the radiation of interest, but also the type of platform being used with the sensor, as well as the required sensitivity. Common sensors include optical sensors such as semiconductors, biological sensors such as the human eye, or thermal sensors.
5. Transmission and processing of the information into a form that is suitable for analysis can occur locally if the sensor is physically connected to a processing computer, or remotely if the sensor information is sent electronically, for example, from a satellite to a ground station.
6. Interpretation and Analysis is done to answer the specific question being asked, whether involving climate change, land usage, vegetation type, vegetation health, and many others.
7. Application is the last step in remote sensing and involves acting based on the interpretation and analysis. This may be simply having an increased understanding of the target through new information, or it may be solving a specific problem.

Remote sensing serves as the means of CO_2 leak detection based on vegetation stress. Specifically, this thesis is centered around developing remote sensing tools and

techniques to monitor the Kevin Dome area, which is part of the Big Sky Carbon Sequestration Project. These techniques will be applicable to other problems as well and are of direct benefit to the hyperspectral community.

Big Sky Carbon Sequestration Partnership

The Big Sky Carbon Sequestration Partnership (BSCSP) is based at Montana State University's Energy Research Institute, and encompasses Montana, Wyoming, Idaho, South Dakota, eastern Washington, and Oregon (Big Sky Carbon Sequestration Partnership, n.d.). One of seven regional carbon sequestration partnerships in the country, BSCSP is supported through a cooperative agreement with the U.S. Department of Energy (DOE). The purpose of the BSCSP is to collaborate with key stakeholders in the identification, development, and monitoring of viable and safe approaches for storing regional carbon dioxide (CO_2) emissions.

The BSCSP is born out of the DOE's Regional Carbon Sequestration Partnership program whose goal is "to help develop the technology, infrastructure, and regulations to implement large-scale CO_2 storage (also called carbon sequestration) in different regions and geologic formations within the Nation." (Regional partnerships, n.d.) Seven regional partnerships were established that encompassed 97% of all industrial and coal-fired CO_2 emissions, 96% of the total land mass, and nearly all the potential sequestration sites in the U.S. Launched in 2003, this initiative had three phases to "develop the infrastructure and knowledge base needed to place carbon storage and utilization technologies on the path to commercialization" (Regional partnerships, n.d.).

Phase I of the DOE's Regional Carbon Sequestration Partnership, Characterization, developed The North American Carbon Storage Atlas (NACSA) (Wright, 2013) that characterized potential CO_2 storage sites in various formations, such as deep oil-, gas-, coal-, and saline-bearing formations, around the United State and Canada. The NACSA, last updated in December 2012, was the guide for choosing test locations in Phases II, and III.

Phase II, Validation, was undertaken to verify sequestration opportunities with testing at small-scales. Nationwide 20 such small-scale tests were performed to demonstrate effective monitoring, develop guidelines for the wells involved, and validate reservoir simulation models. Additionally, public outreach was central as there are numerous stake holders involved. The BSCSP test area was located near Wallula, Washington and was one of the world's first sequestration projects in a basalt formation. The well was drilled in spring 2009 and reached a depth, determined by modeling efforts, of 1253 m. Between July 17, 2013 and August 11, 2013, a total of 977 tons of CO_2 were injected and monitored. Initial monitoring supported simulations as the temperatures and pressure returned to pre-injection levels.

The final phase, Phase III, Development, focused on 8 large-scale storage sites. The BSCSP encompasses the Kevin Dome, a regionally extensive Duperow carbonate formation. The goal is to inject 1 million metric tons of CO_2 into the formation and monitor the geology, geochemistry, water quality, air quality, and underground CO_2 behavior. The Kevin Dome site is in Toole County, Montana, encompassing approximately 1800 km^2. This large area is owned by hundreds of separate land owners showing the necessity of

community outreach including classroom presentations, onsite tours for K-12 students, periodic newsletters, and community appreciation events.

The site has been extensively characterized, both above and below ground. Below ground characterization utilized seismic 'thumper' trucks and geophones to create a 3-D model of the underlying geologic structure. Furthermore, during the drilling of characterization wells both rock cuttings and cores were obtained to measure the underlying geologic structure. Above ground characterization work established baseline measurements of environmental parameters, such as surface and shallow groundwater sampling, soil flux calculations, hyperspectral imaging, and CO_2 differential absorption LIDAR techniques.

Vegetation Stress Indicators

Unlike small-scale carbon sequestration, large-scale sequestration covers thousands of square kilometers; this raises the issue of monitoring these large areas (Rutqvist et al., 2010; Korbøl and Kaddour, 1995; Whittaker, 2004; Maldal and Tappel, 2004). Many CO_2 monitoring options are point measurements that, while useful in areas immediately surrounding injection wells, are not feasible to deploy in large areas for both reasons of cost as well as for land use disruption. Vegetation provides a means of monitoring CO_2 leakage due to how it responds to stress, and more importantly that this stress can be seen in the changing reflectance spectra that can be remotely observed (Bateson et al., 2008; Bellante et al., 2013; Bergfeld et al., 2006; Keith et al., 2009; Maček et al., 2005; Male et al., 2010; Noomen et al., 2008; Noomen and Skidmore, 2009; Pickles & Cover, 1999).

In a simple model of plant stress, three separate reflectance spectral changes can be observed in the range of 350-950 nm (a common range for silicon detectors) at various stages of stress. First, the visible portion, 350-650 nm, of the reflectance spectra will increase caused by the decrease of photosynthetic pigments (Carter et al., 1992; Knipling, 1970). Second, the near-IR portion of the spectra, 750-1100 nm, will decrease due to the changes to the internal structure of the plant (Carter, 1991; Knipling, 1970; Li et al., 2005). Third, a shift is observed is in the red edge region around 700 nm (Horler, 1983) where the spectrum rapidly changes from low in the visible to high in the near-IR. The red edge shifts slightly towards longer wavelengths described in further detail below (Carter and Knapp, 2001; Luther and Carroll, 1999; Eitel et al., 2010; Zarco-Tejada et al., 2000). Due to the sharp rising edge, this change can sometimes be seen before changes are discerned in the visible or near-IR portion of the spectra.

Common green plants share six major photosynthetic pigments (pigments that capture light energy to drive photosynthesis): Carotene, Xanthophyll, Pheophytin a, Pheophytin b, Chlorophyll a (the most common), and Chlorophyll b (Blackburn, 2007; Gates et al., 1965; Lichtenthaler, 1987; Rowan, 1989; Sims & Gamon, 2002). Each of these pigments absorbs in different wavelength ranges, though none absorb past approximately 700 nm. Also, none of the most common pigments absorb well in the green-yellow region, thus the abundance of green seen in nature. As the plant stress increases the relative number of pigments decreases, this decrease in pigmentation means that the plant absorbs less visible light and the reflectance spectra increases in the visible (Abdalla & El-

Khoshiban, 2007; Anjum et al., 2011; Ashraf & Harris, 2013; Jaleel et al., 2009; Tambussi et al., 2002).

Reflection in the near-IR region is due primarily to intercellular spaces and internal scattering caused by cell walls. When a plant undergoes stress the cell walls eventually collapse and the spaces between shrink leading to decreased reflectance. Additionally, photosynthetic pigments do not have absorptions in this region.

The third area of interest is the red edge. This region is the edge of chlorophyll a absorption. As the amount of chlorophyll decreases the red edge moves to higher wavelengths. While the red edge may refer to the region where this transition from low reflectance in the visible to high reflectance in the near-IR, the red edge is customarily defined to be the inflection point of this transition.

Up to this point 'stress' has been used in a broad sense to refer to any type of condition that deviates from the ideal of a plant. Plants can be stressed by a number of different factors including elevated CO_2 in the soil (Bateson et al., 2008; Bellante et al., 2013; Bergfeld et al., 2006; Keith et al., 2009; Maček et al., 2005; Male et al., 2010; Noomen et al., 2008; Noomen and Skidmore, 2009; Pickles & Cover, 1999), water deficiency (Behmann et al., 2014; Dobrowski et al., 2005; Jones et al., 2004; Kim et al., 2011; Peñuelas et al., 1993; Suárez et al., 2008; Tilling et al., 2007; Zhao et al., 2005), nitrogen deficiency (Strachan et al., 2002; Tilling et al., 2007; Zhao et al., 2005), or bugs/parasites/fungi attacking the plant (Apan et al., 2004; Dobrowski et al., 2005; Lawrence & Labus , 2003; Smith et al., 2004; Zhang et al., 2003). Acting simultaneously or sequentially, plants can encounter multiple stresses that result in decreased growth and

internal reallocation of resources leading to changes in the reflectance spectra. However, these differences can be subtle depending on the degree or duration of stress and may not be discernable if the signal to noise of the sensor is insufficient. While the intention of this work was to detect stress due to elevated CO_2 in the soil, insufficient work has been done to quantitatively separate out the types of stress. Furthermore, given that there is no controlled CO_2 release at the Kevin Dome site, nor is there expected to be CO_2 leaks, for the work presented in this thesis this distinction between different stress types is not testable and stress is considered as a whole.

Hyperspectral Imaging

Hyperspectral imaging is a subset of remote sensing and involves recording each pixel of a target with 10's to 100's of narrow spectral bands. Generally, hyperspectral refers to an imaging system that records more than 10 spectral bands with a spectral resolution of 10 nm or less (Shippert, 2007). This is opposed to multi-spectral sensors which may contain 10's of bands, but these bands have a bandwidth of approximately 50-100 nm (Govender et al., 2007; Hall et al., 2002; Vagni, 2007). For example, Landsat is a multispectral imaging platform which delivers 11 bands (with the corresponding bandwidths), but these bands cover the visible (20-75 nm), near-IR (40 nm), short wave IR (100-200 nm), long wave IR (1000 nm), as well as a panchromatic band (180 nm), and a Cirrus band (30 nm) (Knight & Kararan, 2014; Roy et al., 2014)

There are 4 common platforms from which hyperspectral imagery is obtained: satellites, aircrafts, UAVs, and handheld. Several hyperspectral sensors are proposed for

future satellite missions including HyspIRI (HyspIRI Mission Study, n.d.) that would provide world wide data with a 16-day revisit time, similar to Landsat 8, from 380-2500 nm at 30 m spatial resolution as well as 8 bands in the thermal infrared at 60 m spatial resolution with a 5-day revisit time. This mission is still in the study phase, and has the potential to extend the availability of hyperspectral data. Currently, the Hyperion sensor on the EO-1 satellite covers 400-2500 nm in 220 bands at 30 m spatial resolution (Folkman et al., 2001; Pearlman et al., 2001). However, EO-1 does not provide global coverage and must be tasked to study specific areas of interest. Despite this limitation, it has been used in several studies that take advantage of the hyperspectral nature of the sensor (Apan et al., 2004; Galvao et al., 2005; Huete et al., 2003; Kruse et al., 2003; Pengra et al., 2007).

At the other end of the spatial scale from satellites, which routinely deliver 30 m spatial resolution on the ground, hand held sensors (Analytical Spectral Devices Inc., Spectra Vista Corporation, Spectral Evolution, Inc., and others) are a small area measurement, potentially as small as an optical fiber. There are many comparable products that cover the 300-2500 nm range with on the order of 200 bands. These types of sensors are routinely used for ground/plant level verification of remote sensing methods with some providing direct contact measurements with their own illumination.

Both aircraft and UAV are similar platforms and differ in the coverage area and resolution available, but also in the size/weight of the sensors that they can carry. UAVs are suited for small-scale measurements, dependent of fuel/battery, with high spatial resolution, dependent on altitude, making them an attractive proposition for initial research in controlled areas. As the UAV technology improves and the policies are put in place to

maintain safe skies a greater number of systems will be deployed (Weibel & Hansman, 2005). Aircraft platforms span the range from inexpensive to fly single engine aircraft to expensive to fly jet aircraft. Most prominently, the Airborne Visible/InfraRed Imaging Spectrometer (AVIRIS) (AVIRIS Airborne Visible/InfraRed Imaging Spectrometer, n.d.) system developed by NASA has flown on NASA's ER-2 jet, Twin Otter International's turboprop, Scaled Composites' Proteus, and NASA's WB-57. The AVIRIS system is a state of the art system routinely providing data over thousands of square kilometers and is gathering data for the development of the HyspIRI mentioned above. The instrument covers from 400-2500 nm in 224 bands at either 20m or 4 m spatial resolution depending on the aircraft on which it is flown. AVIRIS is a commercial instrument and flights can cost upwards of $70,000, which for many research problems may be far too expensive, especially if multiple flights are required.

Using small aircraft with inexpensive sensors bridges the gap between UAVs and AVIRIS level instruments allowing for studies to be done at the mesoscale. The focus of this thesis work is on areas of a few square kilometers in size with 1m spatial resolution. Using a single engine Cessna aircraft with a commercially available hyperspectral sensor and the processing techniques described herein it is possible to create a mesoscale, radiometrically referenced, high-resolution hyperspectral dataset.

Motivation of Work

To leverage high-resolution mesoscale hyperspectral data, processing and analysis techniques must be developed. The processing should produce high spatial resolution

surface reflectance data that is radiometrically referenced to some trusted standard. Analysis techniques should extend beyond the techniques used for multispectral data as the information is no longer necessarily contained within certain bands, but in the entire spectra. These techniques must also be able to handle the 'big data' associated with having 10's to 100's of spectral bands per pixel. Additionally, from a practical standpoint these techniques must be cost effective, meaning ideally that they should be both automated, to reduce time and manpower, and able to run on a desktop/laptop computer. Furthermore, for the Big Sky Carbon Sequestration project, and other projects or landscapes, the processing and analysis must require minimal (ideally no) site access.

The first portion of the processing presented in the chapter "Hyperspectral Processing" addresses the need of having large area hyperspectral data by taking individual flight passes and radiometrically referencing them to adjacent swaths. This referencing allows the creation of a radiometrically consistent set of data over an arbitrarily large area. Apart from georeferencing, the entire process is automated requiring minimal user intervention and with further software and hardware development could be fully automated. The second part of the chapter deals with obtaining surface reflectance data that is radiometrically referenced to some trusted standard. This is done by using Landsat Surface Reflectance (LaSRC) data that complements the hyperspectral data in the sense that it is a trusted surface reflectance reference that is freely available and has global coverage. Again, this processing is almost fully automated and with minor software development could be integrated into an automated processing structure. Lastly, this

processing does not require any site access for placing tarps or other ancillary instrumentation making it ideal for remote or restricted access sites.

Even though the most appropriate analysis techniques depend strongly on the scientific question being addressed, many questions are poorly answered by simply having a greater number of spectral bands. The novel approach developed in the chapters "Biophysically Relevant Parameters" and "Unsupervised Classification Based on Histogram Splitting" reduces the 80 spectral bands to 9 biophysically relevant parameters and classifies based on these parameters. The parameters are based on a basis function approach that transforms the spectral bands into parameters that correspond to the underlying biophysical structure of the vegetation while also reducing noise. The unsupervised classification is based on a histogram approach to determine similarity as opposed to Euclidian distance utilized in many unsupervised classifications.

This thesis addresses the details of the processing to create mesoscale, radiometrically referenced, high-resolution hyperspectral data and the analysis of such data by fitting individual spectral data with a biophysically relevant model. This model yields biophysically relevant parameters that are analyzed by a novel histogram based unsupervised classification scheme described herein. Lastly, a time series is shown utilizing the radiometrically referenced data.

HYPERSPECTRAL PROCESSING

Data Collection

The primary study area (N48° 51’ 42”, W111° 44’ 13”, elevation 1143 m) is in north-central Montana at the Big Sky Carbon Sequestration Partnership demonstration site, shown in Figure 1. As part of this demonstration project, aerial monitoring of vegetation stress using hyperspectral data will be evaluated as a potential technique for monitoring the efficacy of below-ground CO_2 storage. The use of flight-based spectral imaging will require the direct comparison of data from multiple flights obtained during a single summer and from year-to-year, therefore requiring a method for radiometrically referencing the hyperspectral data. The regions of interest examined herein, outlined in green in the inset of Figure 1, contained regions of fallow fields, planted wheat/barley fields, scrub land, arroyos, draws, and some buildings and roadways. This landscape provided several challenges from the processing point of view because of the myriad of factors contributing to uneven illumination of the landscape, including topography, bidirectional reflectance, and changing cloud cover.

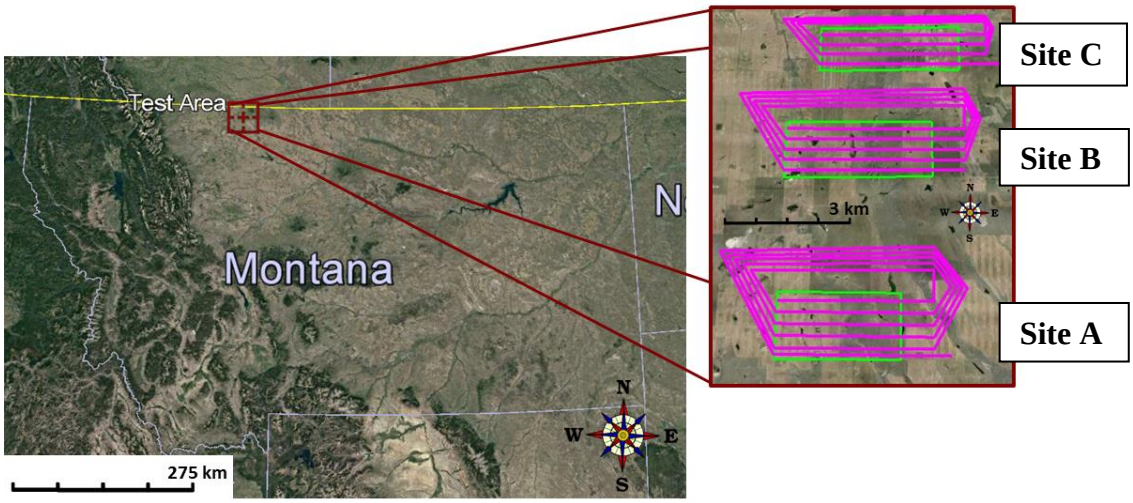


Figure 1: Location of the Big Sky Carbon Sequestration Partnership (BSCSP) demonstration site in north-central Montana. Inset: Map of the 3 test areas (green) and flight paths (magenta) at the demonstration site.

The hyperspectral imaging system used for this data-collection consisted of an imaging spectrometer, a global positioning system/inertial navigation system (GPS/INS), and a flight computer all mounted on a custom-built aluminum plate, shown in Figure 2, and flown in a single-engine Cessna Skyhawk owned and operated by Christopher Boyer of Kestrel Aerial Services, Inc. The imaging spectrometer (Pika II from Resonon, Inc., Bozeman, MT, USA) weighed 1.1 kg, had a spectral range of 425–925 nm, and was binned to provide 80 spectral channels with 6.39 nm spectral resolution. The imaging spectrometer was calibrated in the factory at the time of its manufacture. The GPS/INS (Micro INS, MIS-100-000 from Rockwell Collins, Cedar Rapids, IA, USA) was Wide Area Augmentation System-enabled to improve the tracking accuracy of the GPS navigation and it provided 0.2° pitch and roll accuracy. Hyperspectral data and GPS/INS data were transferred to a P-CAQ PC-104 format flight computer (Resonon, Inc., Bozeman, MT, USA). Flights, shown in magenta in the inset of Figure 1, were made approximately 1220

m above ground level with a 22.5° field of view lens on the dates 06/21/2014, 06/24/2015 and 06/26/2016. Data was taken in a push broom configuration that captured all spectral information in a single instant of time across an entire line perpendicular to the flight direction. Each line captured 640 pixels over 485 m, providing a 0.75 m cross-track resolution. The number of passes was chosen so that, even with the presence of slight turbulence, there would be an overlap between adjacent swaths of 100–150 m. This allowed adjacent swaths to be referenced to each other and guaranteed complete coverage of the test area. Flight passes were flown wherein all were acquired from east-to-west to reduce issues associated with different look angles which produced 4–6 swaths, depending on the area, that needed to be combined/mosaicked to produce the final hyperspectral data. Swaths are numbered 1–N starting in the south and extending to the north corresponding to the order in which they were flown.

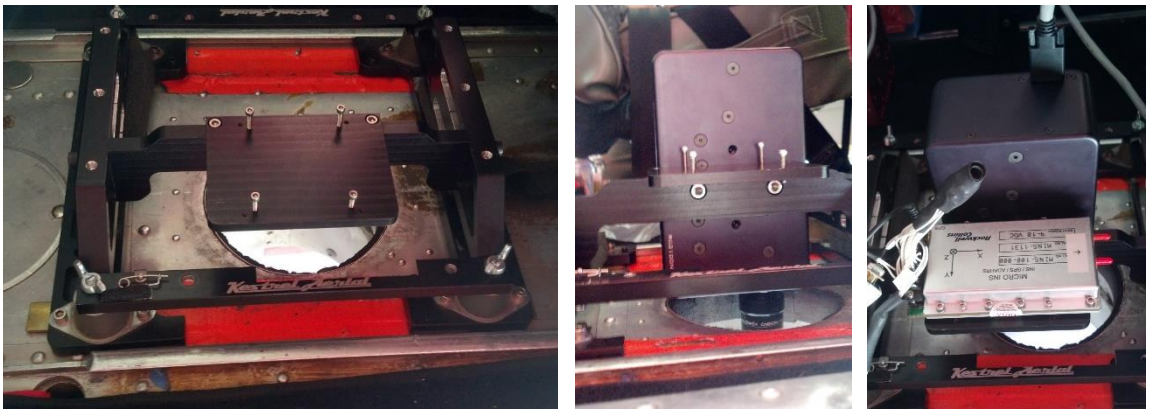


Figure 2: Custom aluminum plate (left) installed in aircraft used for mounting the hyperspectral imager (center) and the GPS/INU (right).

Before the hyperspectral imager, GPS/INU, and flight computer is installed in the aircraft the flight parameters need to be determined and entered into the flight computer.

The altitude of the aircraft and the imaging optics will determine the spatial resolution of the image. To create square pixels using the push broom configuration the air speed of the aircraft and the frame rate of the camera must also be determined. These parameters are determined using "Airborne HSI Calc.exe" provided with the imager. For the flights having 1 m spatial resolution the following parameters are used: Framerate: 55 fps, FOV: 22.5° (0.3937 radians), Imaging Altitude: 4000 ft. (1220 m), Ground Speed: 80 knots (148 km/h). $Imaging\ Altutude$ is determined by the FOV (in radians) of the lens and the desired $Spatial\ Resolution$ by $Imaging\ Altutude = \frac{640 * Spatial\ Resolution}{FOV}$, where 640 is the number of pixels in the camera sensor. The $Ground\ Speed$ of the aircraft is determined by the $Framerate$ of the camera and the desired $Spatial\ Resolution$, $Ground\ Speed = Framerate * Spatial\ Resolution$. The resolutions in both the flight direction and perpendicular to the flight direction are ordinarily the same, as implied here, to yield square pixels.

Additionally, the system requires targets to know when to record data. The targets are defined to be circles inside of which data will be recorded. The targets used are shown in Table 1. These values along with the associated flight lines shown in Figure 1 were determined in Google Earth and uploaded to the flight computer via "ResononGroundStation.exe". Targets are stored in non-volatile memory, while the parameters mentioned above are not and due to the flight computer battery being dead (and difficult to replace) must be entered once the unit is powered up and installed in the aircraft. This also provides an opportunity to set the time based on the GPS signal once the system has been powered up under clear skies for around 5 minutes.

Table 1: Target parameters for Site A, Site B, and Site C, referred to as Area 1, Area 2, and Area 3 respectively. Circles described by these values defines the area in which data will be collected.

Name	Latitude (Decimal Degree)	Longitude (Decimal Degree)	Diameter (Meters)
Area 1a	48.791074	-111.767278	2200
Area 1b	48.791401	-111.751899	2200
Area 1c	48.791244	-111.740178	2200
Area 2a	48.834222	-111.767074	2200
Area 2b	48.834311	-111.748083	2200
Area 2c	48.834132	-111.729332	2200
Area 3a	48.862153	-111.759003	1644
Area 3b	48.861969	-111.751815	1500
Area 3c	48.861877	-111.744098	1600
Area 3d	48.862236	-111.735291	1500
Area 3e	48.862019	-111.728166	1500
Area 3f	48.862029	-111.721129	1500
Area 3g	48.861854	-111.713740	1600

Once the flight is completed data is stored on the flight computer and must be downloaded for further processing. The data is stored as an x-y-λ cube, but the x-y do not correspond to the x-y of the ground so the accompanying GPS/INU data is included as well. Once the data has been downloaded from the flight computer using "ResononGroundStation.exe" processing can begin. The processing steps needed to convert raw hyperspectral data to mesoscale, radiometrically-referenced surface reflectance hyperspectral data are shown in Figure 3. The standard processing steps (Eismann, 2012; Gao, 2008; Thenkabail et al., 2011) that are readily available in most commercial software packages are shown outlined in dashed black, while additional processing steps are outlined in solid blue.

The processing flow can be broken into two separate phases. The first phase of the processing, shown in the upper flow chart of Figure 3, deals with aspects of spatial

processing and the steps necessary to combine raw flight-based hyperspectral data swaths into a self-consistent mesoscale data product. This final data product will be georeferenced to master data wherein individual swaths can be radiometrically-calibrated to match their neighboring swaths. However, this final spatially-calibrated mesoscale data lacks the absolute radiance or reflectance values that are necessary for comparing areas separated from each other in space and/or time.

The second phase of the processing, shown in the lower flow chart of Figure 3, was applied to the spatially-corrected radiometrically self-consistent data, resulting in radiometrically-referenced surface reflectance data. This second phase constitutes removing solar irradiance from the hyperspectral data to obtain surface reflectance values and a subsequent selective spatial and spectral resampling of the data allowing for the hyperspectral surface reflectance data to be referenced to LaSRC data. Solar irradiance was removed using simple atmospheric model and minimizing atmospheric absorption lines to obtain surface reflectance values for the hyperspectral data (See section "Transforming Hyperspectral Data from Radiance to Surface Reflectance"). Spatial processing involved resampling both the LaSRC data and the hyperspectral data to a common resolution so that the reflectance values could be compared on a pixel-by-pixel basis. Spectral processing involved resampling the hyperspectral data based on the Landsat spectral response curves to find what Landsat would measure given the hyperspectral data. After these two processing phases were completed, mesoscale, radiometrically-referenced surface reflectance hyperspectral data were obtained.

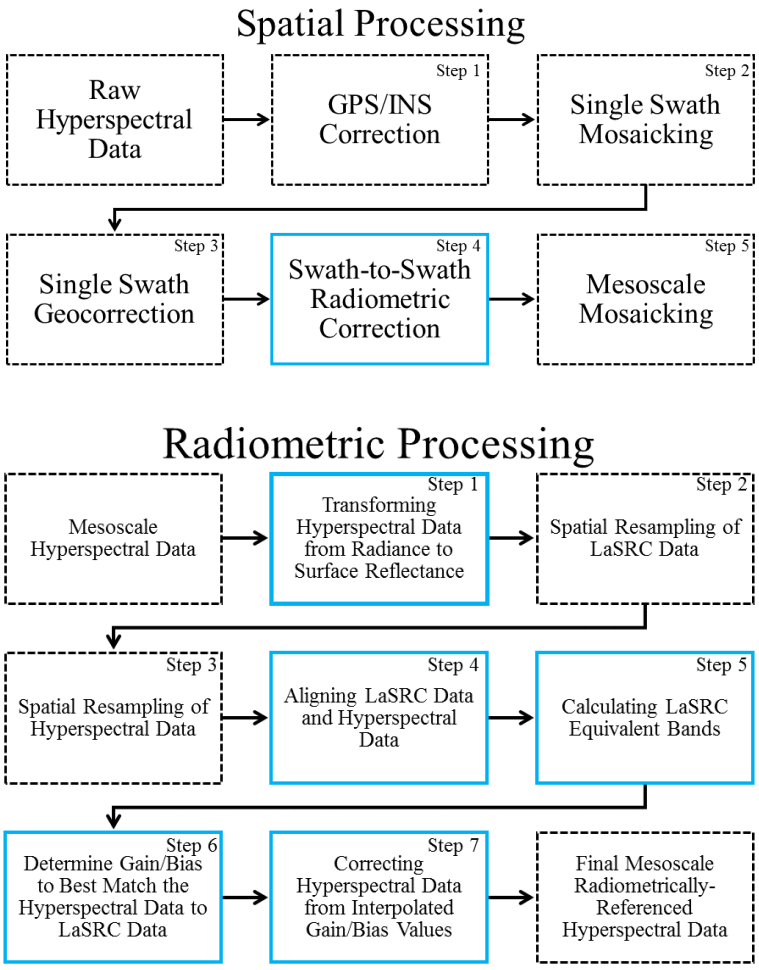


Figure 3: Processing flow chart showing the steps to turn raw hyperspectral data into mesoscale radiometrically referenced surface reflectance hyperspectral data. Steps that extend beyond standard processing techniques (dashed black) are outlined in solid blue. (Upper) Spatial corrections required to assemble mesoscale hyperspectral data. (Lower) Radiometric corrections needed to correct the hyperspectral data to surface reflectance values.

GeoReg – GPS/INU Correction

The raw hyperspectral data, seen in figure 4, along with the GPS/INS information were processed using a post-processing software (GeoReg Resonon, developed for Resonon, Inc. by Space Computer Corporation (SCC), Los Angeles, CA, USA). This

software used the manufacture provided calibration data (640 samples by 80 bands) to calibrate the hyperspectral data to absolute radiance as measured at the sensor, as well as using a vector-tracing algorithm to geocorrect the hyperspectral data to an accuracy of 6–10 m on the ground. This estimated accuracy was primarily limited by the GPS/INS sensor accuracy. Additionally, no Digital Elevation Model (model of elevation across the landscape) was used for this work the software assumed a flat landscape; however, the terrain is relatively flat in this area and since geocorrection is done later this is only a minor issue.

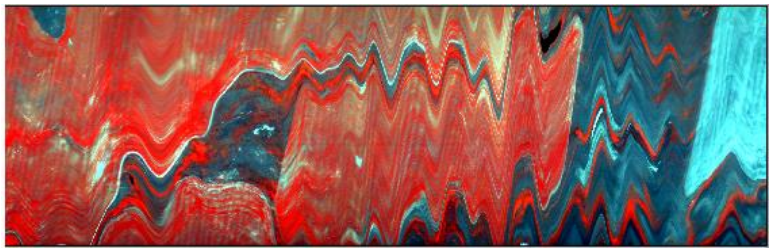


Figure 4: A single block of data taken during a particularly shaky portion of the flight. This shows what happens to the data when the aircraft encounters small amounts of turbulence. Before any type of GPS/INS correction is done the data is unusable. The data is displayed with a standard deviation stretch in false-color IR using bands 10, 20, 70 of the hyperspectral camera which correspond to 490.8, 554.7, 874.2 nm.

File Conversion

The available output file formats were incompatible with the software available and therefore conversion was necessary. The GeoReg program produces files for use with ENVI (Harris Geospatial Solution), one of the geospatial processing programs, which are not compatible with the existing ERDAS Imagine 15 (Hexagon Geospatial) software that was available. Spectronon software (Resonon Inc.) was used to open the *.xv files and

convert them into *.bsq files that can be read into Imagine. While this step was performed manually it is not necessary, and was performed manually due to the software limitations.

Separate Images to Swath Mosaicing

Owing to the data storage technique used in the camera system, it was also necessary to mosaic the separate sections of data stored during a single flight path to obtain a single swath. While recording data the flight computer has a finite buffer and once this buffer is filled the data is then written to disk. This causes a break in the recorded data. The sizes of the areas in this work meant that each flight pass consisted of two such breaks meaning that 3 separate images were produced during each flight pass. The actual break time between the separate images is on the order of 1–2 pixels or less than 50 milliseconds. These images can be mosaicked together with no additional processing or loss of information using the MosaicPro, a module of ERDAS Imagine 15. One such swath, made up of three separate images, is seen in Figure 5.

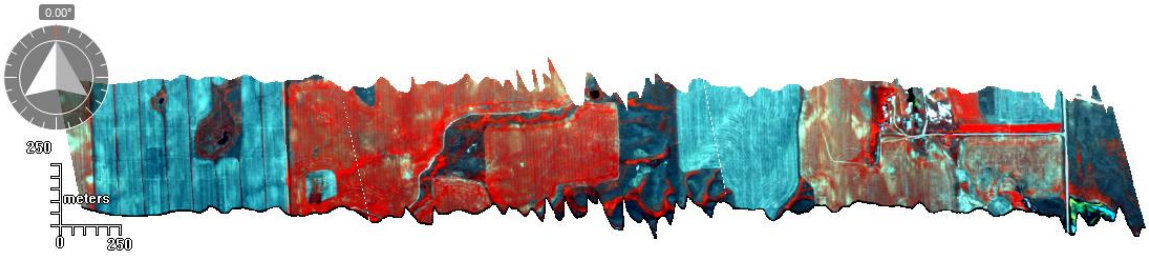


Figure 5: Three separate images mosaicked together after GPS/INU correction. The center chunk was seen before correction in Figure 4. Breaks are seen approximately 1/3 of the way through the image as dashed white areas. The data is displayed with a standard deviation stretch in false-color IR using bands 10, 20, 70 of the hyperspectral camera which correspond to 490.8, 554.7, 874.2 nm.

Single Swath Geocorrection

Georeferencing was performed in AutoSync Workstation, a module of ERDAS Imagine 15 (Hexagon Geospatial), but other commercial software options are also available. Using a freely available U.S. Geological Survey (USGS) 0.3 m resolution orthoimage as master data, persistent landscape features such as road crossings, field intersections, rock cairns and even gully boundaries were found and used as ground control points to further improve the spatial accuracy. The advantage of this approach is that it can be done without the necessity of physically accessing the location to place reference tarps, presuming that master data exists that possesses sufficient resolution and that said data contains persistent landscape features. Each flight path yielded a swath approximately 4400×500 m^2, which is larger than the 485 m width of a single line capture due to roll of the aircraft, and larger than the 4350 m final width due to having a data cushion before and after the test area. For each swath, it was possible to find 40–70 ground control points that could be directly correlated between the hyperspectral data and the master orthoimage. A small example of the types of points that can be used is shown in Figure 6. The ground control points are used to build a polynomial transformation to shift the hyperspectral data from its existing location to the spatially correct location. A zero-order polynomial is used to shift data. A first-order (linear) or affine transformation shifts, scales, and rotates data. Since data is taken in a pushbroom configuration and the pitch, roll, and yaw error will not be uniform across the data higher orders are needed, and since the ground control points are not on a fixed grid, with some being closely spaced and some well separated a simple linear model will be insufficient. While higher order models were tried, they did not

produce an overall root-mean-square improvement as compared to a quadratic model; therefore, a quadratic model was used. This necessitated creating pseudo-ground control points, near the periphery of the images. The points were chosen to follow road or field boundaries as much as possible to constrain the adjustment to a single dimension. The points were iterated until the residuals closely mimicked other nearby points. A limited number of these pseudo-ground control points were created in areas such as agriculture fields where there are few distinguishable features. These points served to remove skew that was present in the image due to imperfect physical camera mounting or from having to fly the aircraft into a slight crosswind. After geocorrection an overall root-mean-square error value of 3–5 m (depending on the swath) was obtained, which is an improvement over the GPS/INS correction. Figure 7 shows the result of georeferencing two separate swaths to the master orthoimage.

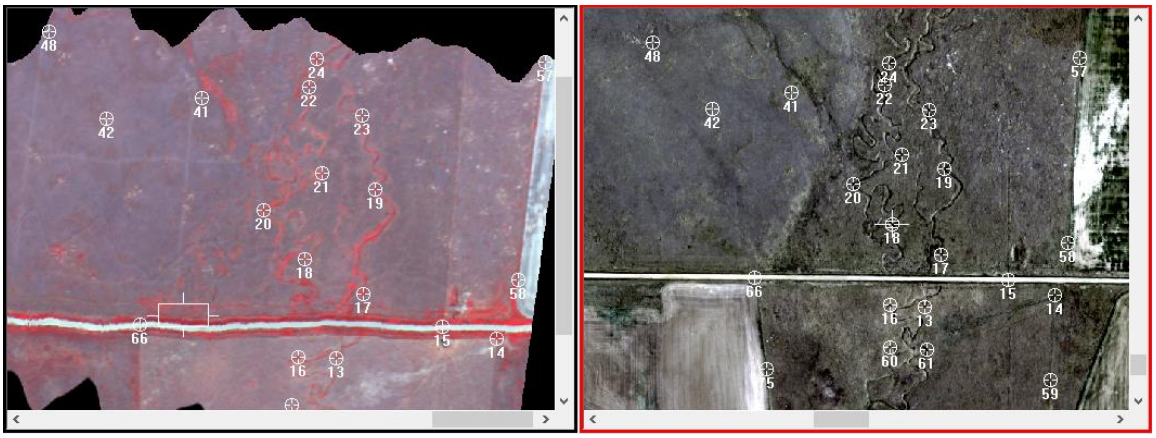


Figure 6: Examples of ground control points which can be correlated between the hyperspectral image (left) and the master orthoimage (right). Usable points include bends in stream in the center of the images, intersections of the road and fields in the lower portion, rock piles at the edges of fields on the right-hand side, and natural landscape variation in the grassland on the left-hand side.

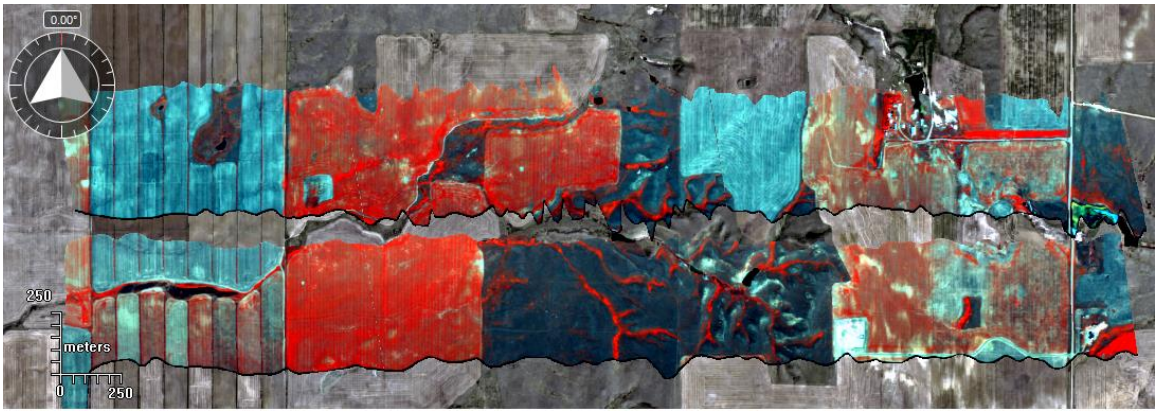


Figure 7: Two separate swaths georeferenced to a USGS 0.3 m resolution orthoimage using at least 40 points per swath. Excellent spatial agreement can be achieved with this type of referencing and can be done without ever having to physically access the test area, making it ideal for remote regions or harsh terrain. The data is displayed with a standard deviation stretch in false-color IR using bands 10, 20, 70 of the hyperspectral camera which correspond to 490.8, 554.7, 874.2 nm.

Coordinate System Conversion

After geocorrection the pixel sizes/shapes do not necessarily align with a set grid, nor is the data necessarily in the correct map projection for comparison to other work. Each swath is reprojected or resampled into the UTM WGS 84 North, Zone 12 (W114–108°) projection with 1 m square pixels. The transformation was done using a Rigorous Bicubic Spline transformation, which is computationally intensive and time consuming. Additionally, during this step the data is saved as a GeoTIFF that can be natively read into Matlab.

Swath to Swath Radiance Correction

The primary radiometric calibration issue to be dealt with before swaths can be mosaicked is the effects of changing illumination conditions between flight passes.

Lighting conditions may change between flight passes for many reasons, such as high-altitude clouds, movement of the sun and imperfect auto-calibration of the hyperspectral imaging camera's shutter speed (which might be necessary to avoid saturated pixels). Two adjacent geocorrected swaths (first, southernmost, and second flight passes) were mosaicked together with no radiometric calibration are shown in Figure 8. It can be seen in these data that the northern portion is darker than the southern portion and that this difference is present throughout, though less pronounced on the west side. During this data-collection flight, the pilot reported observing high-altitude cirrus clouds (Vaughan et al., 2010) drifting over the study area, which supported conclusion that these clouds were the cause of the illumination difference. This discrepancy could be reduced by the addition of an upward-facing illumination sensor, but such a sensor is still a point measurement and may therefore miss lighting artifacts near the edges.

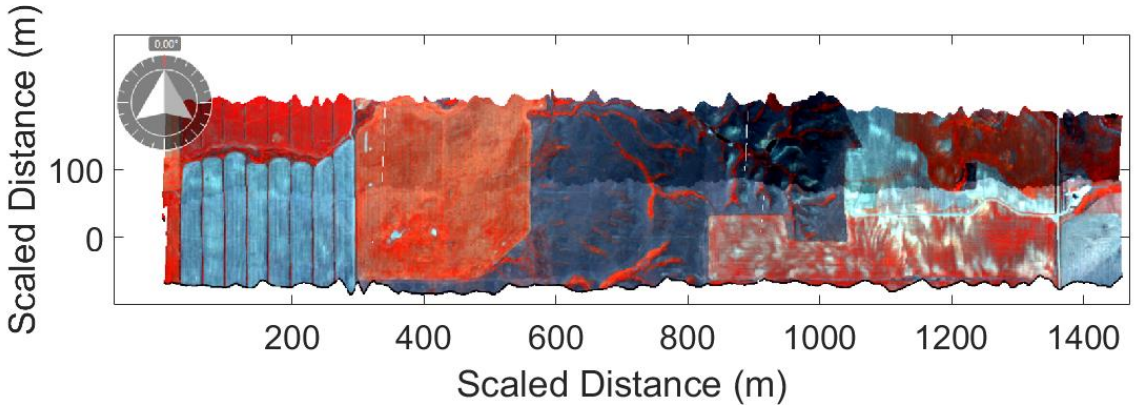


Figure 8: Two adjacent geocorrected swaths (1 and 2) mosaicked together with no radiometric calibration. The swaths show the differing global illumination conditions between them, and the changing lighting conditions along the northern swath that are due to high-altitude clouds. This illumination difference makes it impossible to have a self-consistent data when mosaicking multiple swaths with different illumination conditions. Scaled distances correlate with Figures 8–11. The mosaic is displayed with a standard deviation stretch in false-color IR using bands 10, 20, 70 of the hyperspectral camera which correspond to 490.8, 554.7, 874.2 nm.

Figure 9 shows, for each overlapping pixel, the absolute difference (AD) between the two adjacent swaths shown in Figure 8. The AD is the absolute value of the radiance difference between the *n*th band in the North and South swath summed over all 80 bands, such that

$$AD = \sum_{n=1}^{80} |North(n) - South(n)|, \quad (1)$$

where North(*n*) and South(*n*) are the radiance values (in μflicks [μW sr^{-1} cm^{-2} μm^{-1}]) of the *n*th band in the Northern and Southern swaths, respectively.

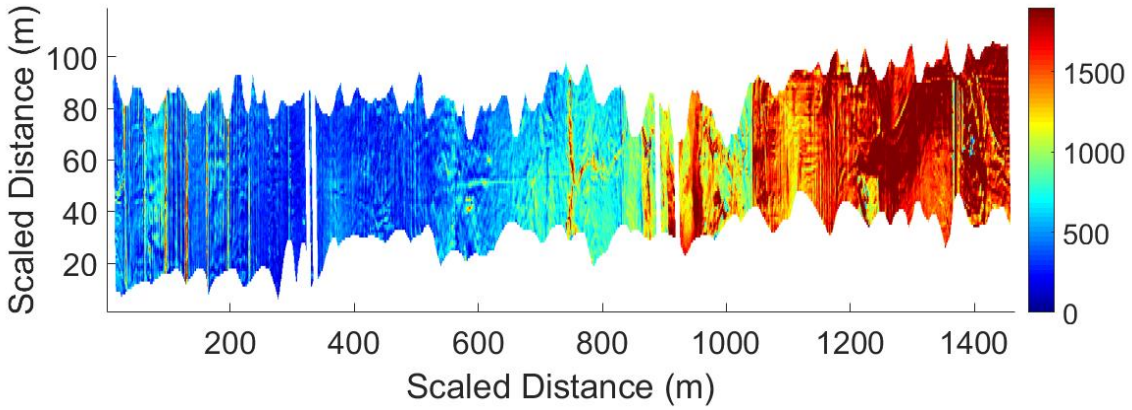


Figure 9: The total absolute difference (AD), in μflicks shown as a color plot, at each pixel in the overlap region between swaths 1 and 2 shown in Figure 8. The mean offset per pixel per band is ~960 μflicks, a significant difference when standard pixel values are on the order of 5000–6000 μflicks. The east (right) side if the image has a larger AD due to clouds shading only part of swath 2.

As expected from the false-color image (Figure 8), the pixels on the east side have a larger AD than those on the west due to the cloud cover shading the east side and not the west causing a greater radiance difference between the northern swath and the evenly illuminated southern swath. To remove slight misalignment issues owing to imperfect geocorrection and to save memory during processing, the overlap area was downsampled

by a factor of 3 using the Matlab 'imresize' command utilizing the default bicubic interpolation method (see Appendix A). This resizing reduced the memory requirements by a factor of 9 and improved processing time by at least the same factor.

On a pixel-by-pixel basis, the 80 hyperspectral bands of the southern swath (here chosen to be the master data), swath 1, are compared to the corresponding 80 bands of the northern swath, swath 2. Fitting a linear model to these data points yielded the gain and bias that corrects the radiance of the northern swath to that of the southern swath. Example pixels are shown in Figure 10. This linear model was justified by the fact that the sensor is linear in intensity, regardless of wavelength, if it is not saturated. With these gain/bias values it was possible to modify each pixel in the northern swath to match the southern swath (to within the sensor noise) in the overlap area. It was then necessary to generalize the results to correct the entire northern swath.

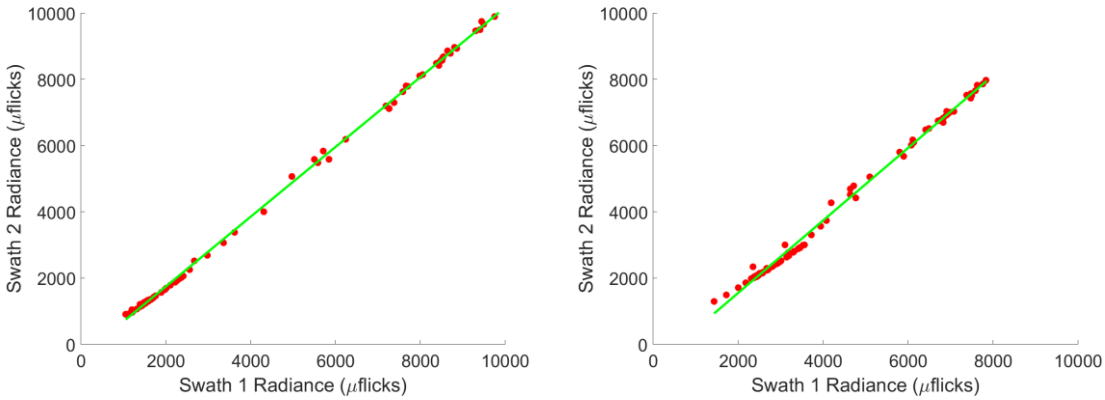


Figure 10: Two examples of a single pixel of swath 1 plotted against the pixel from swath 2 at the same spatial location (red dots), and the corresponding linear fit (green) yielding the gain and bias necessary to correct pixel in swath 2 to match swath 1.

To correct the northern swath, it was assumed that the illumination in an entire cross-track column (vertical line of pixels) was homogeneous and therefore could be

corrected by the same gain and bias as those within the overlap region. Therefore, the gain and bias for each pixel in a cross-track column were averaged (vertical direction in Figure 9) and are plotted Figure 11. These average gain and bias values were used to correct the radiance values of the northern swath to match the southern swath when applied to each cross-track column, where a perfectly-matched illumination (and therefore radiance values) between adjacent regions would give a gain value of 1 and a bias of 0. Differences from these ideal values indicate differing illumination conditions between the regions. When observing a specific area under different illumination conditions, illumination variation will manifest as a difference in gain because the sensor is linear in intensity. Variation from an ideal bias is primarily owing to imperfectly-matched pixels between the two swaths, which is most easily seen in Figure 11 as the spikes on the west side of the image. These spikes are likely due to a slight misalignment between the northern and southern swaths and if this is the case are pixels containing vegetation being compared to pixels containing bare ground and vice versa. Similar issues can be seen at other field boundaries such as that around 600, where different vegetation coverages are being compared between the two swaths. These artifacts can be reduced with improved geocorrection. Owing to these artifacts and using the assumption that illumination conditions vary smoothly over the area, a smoothing filter was employed. The 'smooth' command in Matlab was used with the 'rloess' method, which is a local regression that uses weighted least squares and a second-degree polynomial that assigns zero weight to the data outside of six mean absolute deviations. A 201 pixel-wide window was used, which with the resizing factor is the same scale as the widths of the field in the sample area. Finally,

the smoothed gain and bias were applied across the entire swath. This corrected for the illumination disagreement between the two swaths along the flight direction.

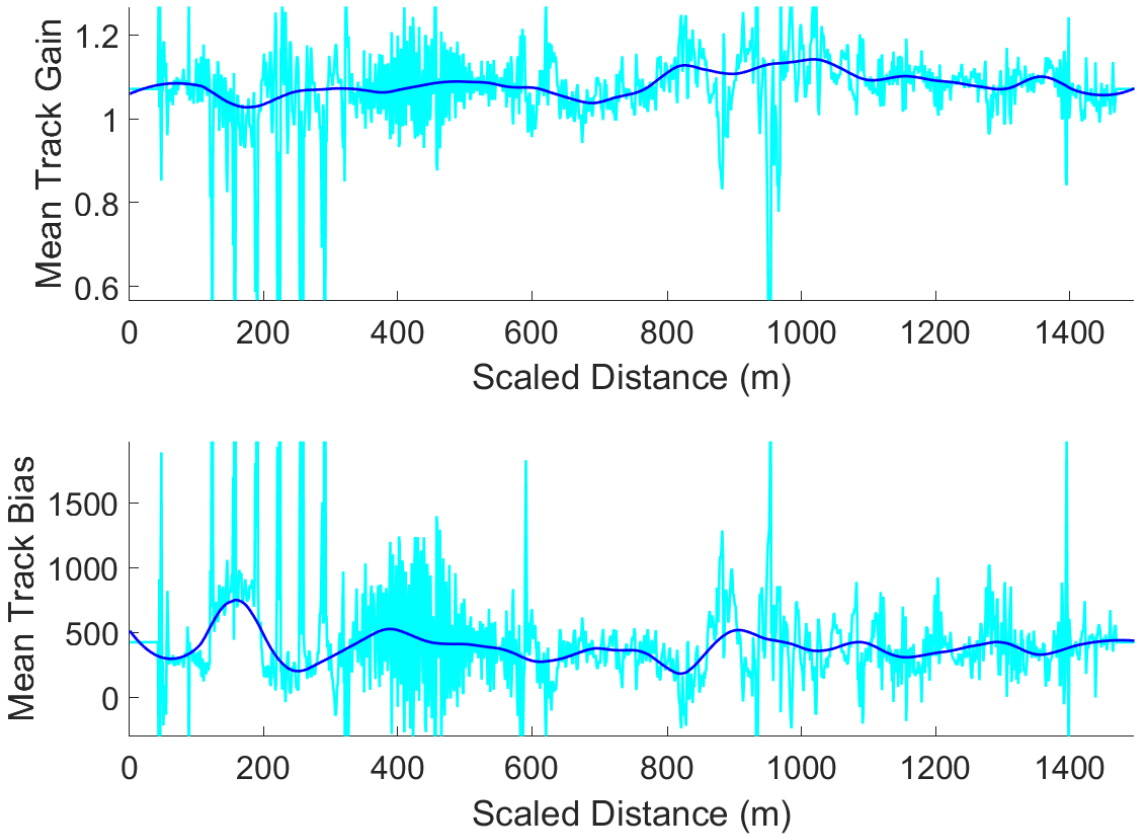


Figure 11: Gain (Upper) and bias (Lower) values necessary to match the corresponding pixel in the northern swath to that of the southern swath averaged along the cross-track (vertical) direction. If the two swaths had identical illumination conditions the gain would be exactly 1 and the bias would be 0, which is nearly the case in the regions where illumination was similar between the swaths (left hand side).

To further correct the northern swath to match the southern swath, a cross-track adjustment was made assuming a simple linear model. This cross-track adjustment was made in the vertical direction and helped to remove slight vignette artifacts originating from the camera. While the cross-track adjustment was a minor change, it contributed to the overall agreement between the swaths. The adjustment had a slight tendency to change

the overall illumination of the northern swath, however, and it was therefore necessary to complete a swath-wide illumination correction as a last step. These two corrections (here denoted as cross-track and illumination) were implemented in Matlab using the 'fminsearch' command with a custom input function (see Appendix A). This function allowed for either a cross-track or illumination adjustment and returned a single figure of merit given by the total sum over all pixels of the AD between the northern and southern swath. When ‘fminsearch’ minimized this figure of merit, the radiometric agreement between the swaths was maximized. The resulting parameters were applied to the entire northern swath and the resulting corrected image is shown in Figure 12, whose vertical scale is the same as that in Figure 9. There are still regions of mismatched illumination in the image, owing to bidirectional reflectance originating in areas with uneven terrain, such as gullies on the east side, and areas possessing a slight spatial mismatch, as shown in the fields on the west (left) side. The average offset per pixel per band is reduced from 957 μflicks (before correction) to 265 μflicks (after correction), which is a factor of 3.6 improvement. The average offset per pixel per band of 265 μflicks after correction is on the order of the variation within a single agriculture field, 226 μflicks, or uniform grassland, 204 μflicks, and therefore may not be improved regardless of technique.

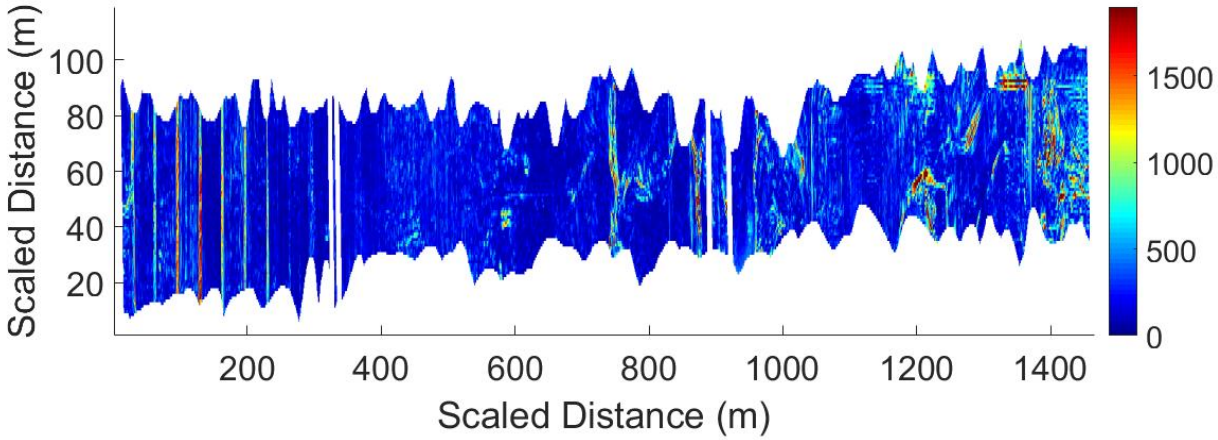


Figure 12: The same overlap area in Figure 9 after using the process to match the illumination (track and cross-track) between the two swaths. The mean offset per pixel per band is ~260 μflicks, which is a marked improvement over Figure 9 and is similar to variance seen in uniform areas of the data.

The overlap matching technique was fully automated and required no user input. This relative radiometric processing was continued across an entire test area to process as many swaths as may be required. This processing step creates a set of swaths that are radiometrically consistent.

Full Area Mosaicking

The individual swaths were mosaicked in MosaicPro. Seamlines in the image were generated based on the default values associated with the Weighted Seamline routine. If areas were missed due to turbulence, the seamlines were adjusted manually to include these areas.

Gap Interpolation/Extrapolation

To remove the gaps present in data collection, a routine in Matlab was used to interpolate across the gaps on a band-by-band basis. A Matlab routine based on directly solving a system of linear equations for ∇^2 across the gaps was used (see Appendix A). After these processing steps, a complete dataset of the entire test area was generated and is shown in Figure 13. This method of interpolation works well for gaps of a few pixels, but does not work as well in large areas such as the eastern portion of Figure 13 where a sizable portion of the data is missing. These data can now be analyzed as an entire unit of self-consistent radiometric data instead of four separate swaths possessing their own distinct radiometric ranges. However, these data lack any absolute radiometric values, making it difficult to compare to other areas or to develop temporal trends in a quantitative manner.

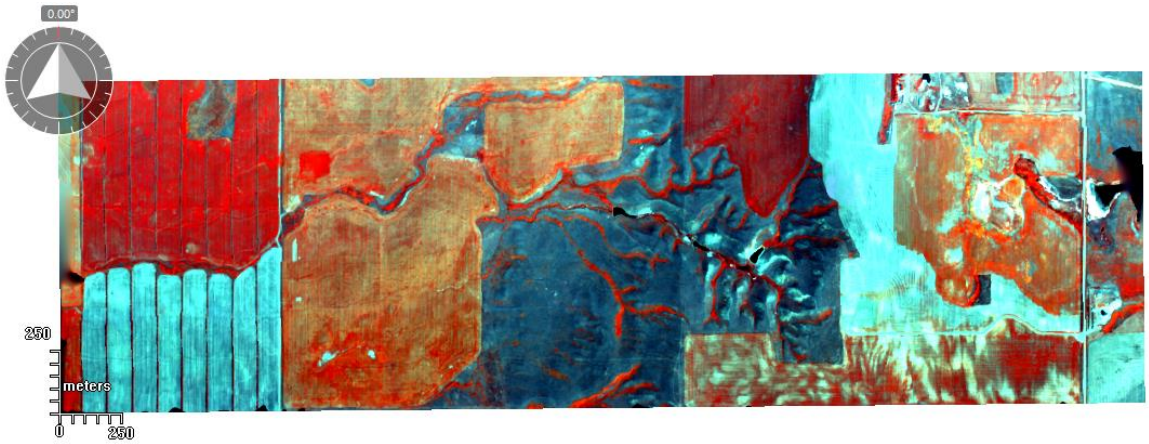


Figure 13: A full dataset consisting of 4 separate swaths mosaicked together after swath to swath radiance correction. The image is self-consistent across the entire area but is lacking any absolute radiance standard. The mosaic is displayed with a standard deviation stretch in false-color IR using bands 10, 20, 70 of the hyperspectral camera which correspond to 490.8, 554.7, 874.2 nm.

Transforming Hyperspectral Data from Radiance to Surface Reflectance

To convert the at-sensor radiance values, it is necessary to divide them by the solar irradiance as attenuated by the atmosphere to obtain surface reflectance data. A number of sophisticated models for this process exist, including MODerate resolution TRANsmission model (MODTRAN) (Berk et al., 2005), Atmospheric/Topographic CORrection (ATCOR) (Richter & Schlapfer, 2012), Atmospheric COrrection Now (ACORN) (Green, 2001), Fast Line-of-sight Atmospheric AnalysiS of Hypercubes (FLAASH) (Cooley et al., 2002), and Simple Model of the Atmospheric Radiative Transfer of Sunshine (SMARTS) (Gueymard, 2005), any of which could be used to calculate the atmospheric attenuation of the solar irradiance for calculation of surface reflectance. For simplicity, however, a model from ASTM G173-03 standard was chosen, which contains Reference Solar Spectral Irradiance values (Standard, 2007). This standard contains data for both the extraterrestrial (air mass 0) and terrestrial reference spectra at air mass 1.5 as calculated for a standard atmosphere and a rural aerosol profile, which is appropriate for the areas under investigation. With these two sets of data and by assuming a linear relationship between them, it was possible at each wavelength to simulate any arbitrary air mass. To determine the appropriate air mass the hyperspectral data is divided by the solar irradiance and the O_2 absorption features observed in the hyperspectral data near 760 and 820 nm are removed (Appendix A). This is accomplished by comparing the spectrum to a smoothed version of the spectrum and minimizing the difference.

Because such a large area was being examined, it was necessary to choose a subset of the data as a representative spectrum. Based on a simple normalized differential

vegetation index (NDVI) (Carlson & Ripley, 1997; DeFries & Townshend, 1994; Wójtowicz et al., 2016) measurement, the top 15% most vegetative pixels (highest NDVI values) were chosen which, after averaging, gave a representative spectrum with very low noise that could then be used to determine the air mass necessary to best remove the O_2 absorption features. Given the strong absorption features present, it was also possible to calibrate the wavelengths of the hyperspectral channels to improve the accuracy of the data both in terms of wavelength and surface reflectance. Once the air mass was determined, the entire area was scaled by dividing by the irradiance, which thereby yielded a surface reflectance data that could be directly compared to the LaSRC data. Results of this simple atmospheric model are shown in Figure 14, which uses the average of the top 15% most vegetative pixels for both the extraterrestrial solar spectrum and the derived air mass spectrum. In the derivation, the result of an air mass value less than 1 is owing to several factors. One of these factors is that the sensor was not located at ground level, and therefore the direct and scattered irradiance was different than the ASTM standard (Standard, 2007). Another factor is that the elevation of the site was 1000 m above sea level and therefore possessed different H_2O and O_2 columns than those of the standard.

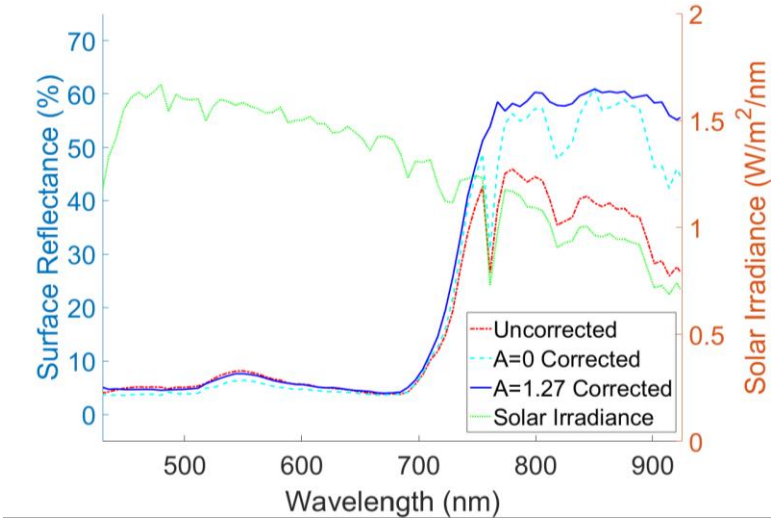


Figure 14: Plot showing the uncorrected (scaled by mean value of extrasolar irradiance for display purposes) hyperspectral data (dot-dashed red curve), and the data corrected using the extraterrestrial solar irradiance (A=0) (dashed cyan curve), the derived solar irradiance (A=1.27) (solid blue curve) and the actual solar irradiance used for correction (dotted green curve). The derived solar irradiance removes absorption features quite effectively.

LaSRC Data Product

The radiometric data from Landsat 8 was used as the reference standard owing to its quality of data in terms of calibration, signal-to-noise, global coverage, and quality of processing including atmospheric modeling (Barsi et al., 2014; Knight & Kararan, 2014; Markham et al., 2014; Roy et al., 2014; Storey et al., 2014). The Provisional Landsat Surface Reflectance Product (LaSRC) data is a high-level data product freely available from the USGS. Atmospheric modeling is done on the data before delivery removing the necessity of the end user to do the modeling. Aerosol, Ozone, water vapor, and air temperature are taken from MODIS Climate Monitoring Assessment (CMA) data and an internal radiative transfer algorithm is employed to model the atmosphere and obtain surface reflectance from the at sensor data.

Matching Hyperspectral Data to LaSRC Data

Using LaSRC as a radiometric reference target, it was possible to match the surface reflectance hyperspectral data to a trusted radiometric source despite the spatial, spectral, and reflectance differences between the two datasets (Appendix A).

Spatial Resampling of LaSRC Data and Hyperspectral Data

Initial processing was performed by ERDAS Imagine that used Rigorous Cubic Convolution to re-project the LaSRC data at 30 m resolution into the WGS 84 / UTM zone 12N projection, which is the same projection as that of the hyperspectral data. These re-projected data were then used as a radiometric reference for the hyperspectral data. Ideally, hyperspectral flights would be coincident with Landsat data collection to remove the possibility that changes in the atmosphere or landscape would affect the data, but compromises must be made when coincident data collection is not possible.

To compare the hyperspectral data with its 1 m resolution to the 30 m resolution LaSRC data, both datasets were resampled via Matlab to a common spatial resolution of 7.5 m using bicubic interpolation. The spatial resolution is a compromise between upsampling the Landsat data, to blur the large square pixels, and downsampling the hyperspectral data and creates a common framework within which it is possible to compare the two datasets on a pixel by pixel basis. The 1 m spatial resolution of the hyperspectral data was not abandoned but was temporarily reduced to allow direct comparison.

Spatial resampling is not a new idea and major ideas in this work were borrowed from digital photography. Advanced resampling techniques such as edge sampling (Fattal,

2007), new edge-directed interpolation (Li & Orchard, 2001), error-amended sharp edge (Cha & Kim, 2007), soft-decision adaptive interpolation (Zhang & Wu, 2008), and Gaussian radial basis function (Lee & Yoon, 2010) could improve the spatial quality of the final data (Zhao et al., 2010). The best technique may be landscape dependent, however, and care should be taken in choosing resampling algorithms.

Aligning LaSRC Data and Hyperspectral Data

Due to the imperfect georeferencing, terrain features can be shifted between data sets. Therefore, it was necessary to spatially shift one set of data to match the other. This was accomplished via Matlab using the built-in image transformation commands to determine the geometric transformation that maximizes spatial alignment between two 2D (two-dimensional) grayscale images (Appendix A). The grayscale images were obtained directly from the red, green, and blue channels of the LaSRC data and from the equivalent three channels of the hyperspectral data after it was converted into the equivalent LaSRC bands (described below in "Calculating LaSRC Equivalent Bands" section). The grayscale hyperspectral image was registered to the grayscale LaSRC image in Matlab using the 'imregtform' command, constrained to only allow spatial translation. The resulting shifts were on the order of 10 m, which were less than the 30 m resolution of the LaSRC data and were therefore reasonable.

Calculating LaSRC Equivalent Bands

The hyperspectral camera covers the spectral wavelength range 425–925 nm, within which six Landsat bands are contained. These Landsat bands include Coastal Blue,

Blue, Green, Red, IR, and the Panchromatic band. The Panchromatic band is not used in the processing as it is not included as part of the LaSRC data. The spectral response of these Landsat bands as well as a pair of adjacent hyperspectral bands are shown in Figure 15. Each Landsat band covers multiple hyperspectral bands, with the Coastal Blue band covering 6 hyperspectral bands and the Green band covering 16 hyperspectral bands, while some hyperspectral bands are outside the range of the Landsat bands. Using Landsat spectral response curves, it was possible to spectrally resample the hyperspectral data to directly correspond to the Landsat bands, which thereby allowed direct comparison between the hyperspectral and LaSRC data. A single Landsat equivalent channel (LS_{Equiv}) was obtained by multiplying the full spectral response curve (LS_{Resp}) by the hyperspectral values at each wavelength ($HS(n)$) and by summing these values over all 80 bands. Lastly, the LS_{Equiv} must be scaled by the ratio of the spectral width of the hyperspectral channels (fixed at 6.39 nm for this camera configuration) to the full width half max (FWHM) of the appropriate Landsat channel (LS_{FWHM}). This is represented by

$$LS_{Equiv} = \left(\sum_{n=1}^{80} LS_{Resp}(n) * HS(n)\right) * {}^{6.39}/_{LS_{FWHM}}, \quad (2)$$

which gives the surface reflectance that a specific Landsat channel would see if it was observing the same spatial location observed by the hyperspectral camera.

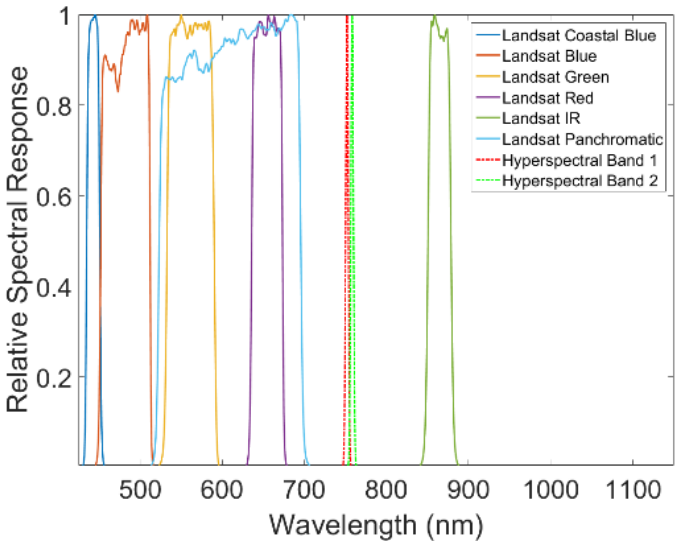


Figure 15: Relative Spectral response curves of the Landsat satellite (solid curves), and two adjacent bands from the hyperspectral camera (dashed curves).

Determining the Gain/Bias to Best Match Hyperspectral Data to LaSRC Data

The registered 7.5 m resolution LaSRC data was compared on a pixel-by-pixel basis to the corresponding 7.5 m resolution hyperspectral data pixel. An additional degree of freedom was provided wherein the hyperspectral data in each pixel could have a gain/bias applied in intensity space before being converted to its scaled Landsat equivalent,

$$LS_{Equiv,Scaled} = \left[\sum_{n=1}^{80} LS_{Resp}(n) * (Gain * HS(n) + Bias)\right] * {}^{6.39}/_{LS_{FWHM}}. \quad (3)$$

A linear model was used for the gain/bias because both sensors possessed a linear response in intensity, though they did not necessarily have the same responsivity or dark current. By changing the gain and bias for each pixel it was possible to minimize the difference between the scaled hyperspectral data and the LaSRC data, and to thereby radiometrically correct the hyperspectral data, using the LaSRC data as a reference, while preserving the information contained in the ratio of individual hyperspectral bands. To minimize the

difference between the LaSRC data and the scaled hyperspectral data, the 'fminsearch' command in Matlab was used to search for the best gain and bias. For this it was necessary to write a custom Matlab function (Appendix A) that took the hyperspectral data, the LaSRC data and the Landsat spectral response data as inputs and, after applying the gain and bias to the hyperspectral data, calculated the equivalent LaSRC response. This was identical to the processing described in "Calculating LaSRC Equivalent Bands" with the addition of applying the gain and bias to the hyperspectral data before calculating the LaSRC equivalent. The sum of the squared difference between the LaSRC data and the LaSRC equivalent data was weighted on a band-by-band basis based on the number of hyperspectral bands covered by a particular Landsat equivalent band. These weights were 6 for the Coastal Blue, 16 for the Blue, 16 for the Green, 10 for the Red, and 14 for the Near-IR. This weighting gave a higher weight to bands covering a larger wavelength range and lower weight to bands such as Coastal Blue, which only overlaps with 6 hyperspectral bands. This number was minimized to determine the gain and bias that most precisely matched the hyperspectral data to the LaSRC data.

Correcting Hyperspectral Data from Interpolated Gain/Bias Values

Once the best gain and bias values were determined for each pixel (using the 7.5 m spatial resolution) the entire grid of gain and bias values was resized to the dimensions of the original data (i.e., back to 1 m spatial resolution) using 'imresize' with bicubic interpolation. The Interpolated gain and bias values for the entire area are shown in Figure 16. The gain varies between 0.5 and 1.1 for most of the image and is smoothly varying. The bias varies between -500 and 500 and with the scaling used (0–100% covering a range

of 0–10,000 as LaSRC data does) this range corresponds to ±5%. The interpolated values were then applied to each pixel of the hyperspectral data as a linear global spectral correction to create mesoscale radiometrically-referenced surface reflectance hyperspectral data, as shown in Figure 17. While the data in Figure 17 appears to have a degraded spatial resolution, upon closer examination in Figure 18 it becomes obvious that the spatial resolution is unchanged and details emerge which were not readily visible before correction. However, in the fields in the left center of Figure 13, the local contrast is reduced leading to the overall appearance of blurring or loss of spatial resolution.

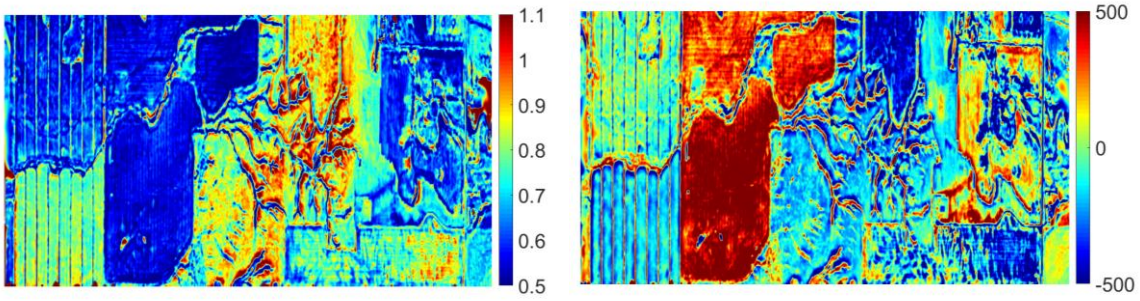


Figure 16: Interpolated gain (left) and bias (right) determined by minimizing the difference between LaSRC data and the equivalent hyperspectral data. Values are smoothly varying across the image so local contrast is maintained.

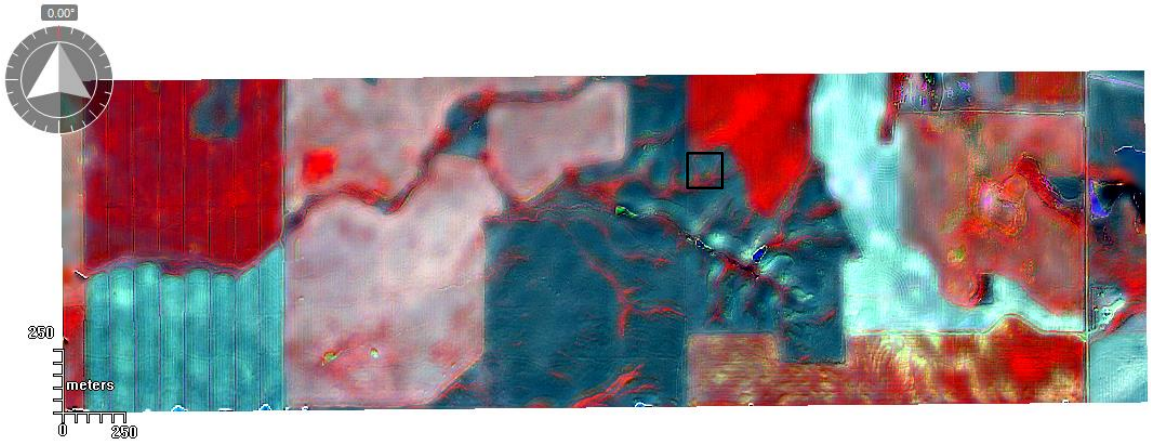
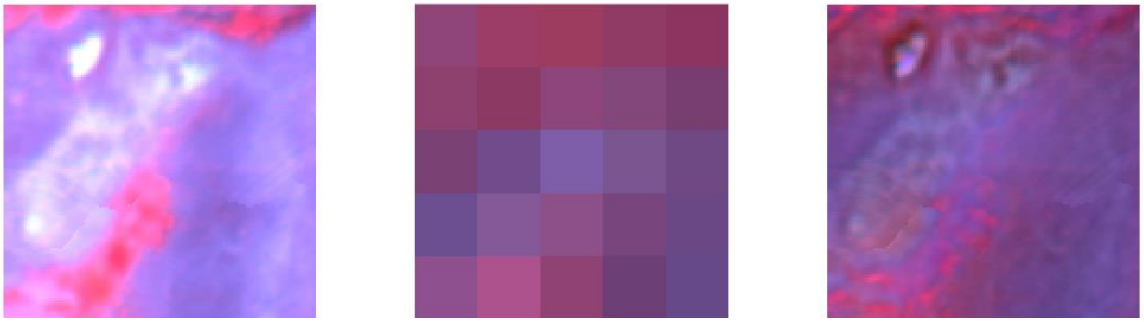


Figure 17: The hyperspectral data after surface reflectance correction of five Landsat bands. Because the radiometric correction was performed with coarse resolution data, some of the small-scale saturation has been reduced giving a blurred appearance. The region in the black box is magnified in Figure 18 showing the spatial resolution is maintained after processing. The data is displayed with a standard deviation stretch in false-color IR using bands 10, 20, 70 of the hyperspectral camera which correspond to 490.8, 554.7, 874.2 nm.



Before Correction | LaSRC data | After Correction

Figure 18: All 3 images used the same manual dynamic range adjustment to display false-color IR images displayed using the average of bands 5–15, 15–25, 65–75 of the hyperspectral camera before correction to LaSRC (left), and after correction (right) and the false-color IR (bands 2, 3, 5) image of the LaSRC data (middle). The region encompasses 25 Landsat pixels (5 x 5 grid measuring 150 m x 150 m) and the corresponding 22,500 hyperspectral pixels. The spatial resolution of the hyperspectral data is not degraded, though the local contrast is reduced in some areas due to the spatial resampling. Image location is shown in Figure 17 for reference.

Mathematical Validation

To spatially verify the accuracy of the corrections to the hyperspectral data, 2000 areas were chosen at random possessing a range of sizes from 1–1681 m^2. The average values of the LaSRC data and the hyperspectral data (after being transformed into the LaSRC equivalent) in an area were then compared on a band-by-band basis. From these values, a mean absolute percentage error (MAPE) value was obtained for various area sizes, as shown in Figure 19 (Appendix B). Despite the diverse sizes of the random test areas, the agreement of their MAPE value was very consistent at all scales even though 30 m resolution data were used to correct 1 m resolution data. This comparison within the corrected data shows the significant agreement between the hyperspectral data and the LaSRC data in a quantitative sense.

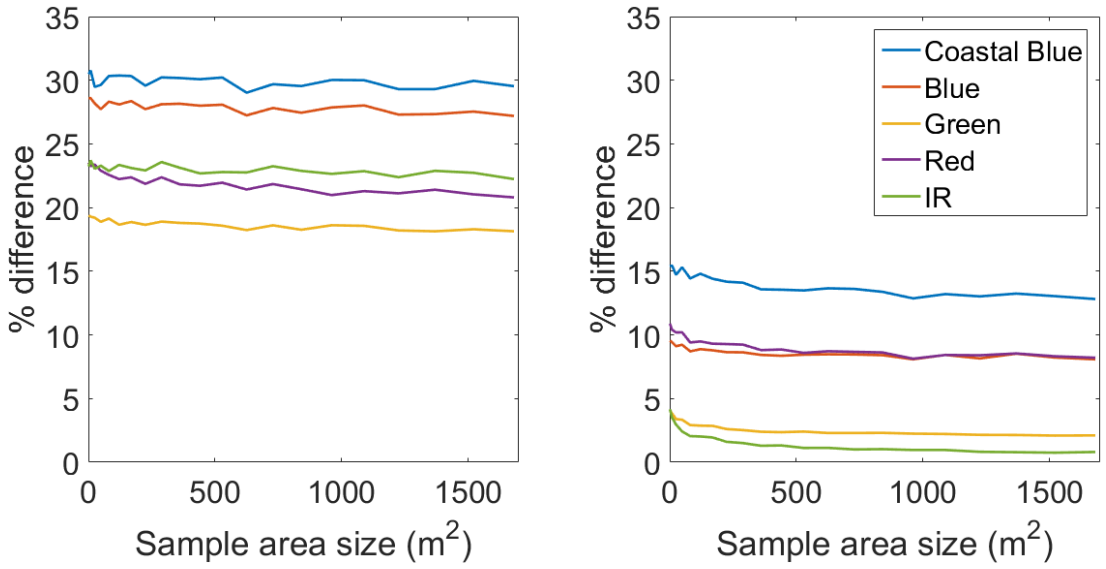


Figure 19: Mean absolute percentage error between the uncorrected (left) and corrected (right) hyperspectral data compared to LaSRC data averaged over 2000 random areas of varying sizes.

Wheat Field Validation

Since it was not possible to access the site under study to collect data on the surface reflectance, a similar agricultural site was used instead to verify the spectral accuracy of this technique. A winter wheat field located outside of Bozeman, MT (N45° 33' 41', W111° 10' 26", elevation 1461 m) was flown on 06/02/2016 with the same imaging system described above, but at an altitude of 600 m yielding 0.5 m pixels. Data was processed using the technique described in this thesis with LaSRC data from 6 days after, 06/08/2016. Additionally, data was collected with an ASD Field Spec Pro 350 on the wheat in the field on the same day as the flight. The ASD is a 16-bit spectrometer that has a spectral range of 350–2500 nm. The sampling interval for the instrument is 1.4 nm at 350–1000 nm and 2 nm at 1000–2500 nm.

ASD data was acquired for wheat at 17 separate locations around the Bozeman, MT test area and a representative spectrum was determined. This representative spectrum was used for comparison to 25 random pixels throughout the field. The test area was a uniformly managed wheat field and was expected to be uniform in terms of reflectance spectra and this was seen when examining the 25 random pixels. A comparison of two of these pixels (each 0.5 x 0.5 m) demonstrates the ASD data relative to the hyperspectral data before and after correction to LaSRC data as well as the actual LaSRC data at this same location is shown in Figure 20, as well as the average error percent difference between the ASD data and the hyperspectral data for the 25 random pixels. The average percent difference over the 25 test areas was reduced from 4.16% before correction to 2.74% after correction to the LaSRC data. The agreement between the LaSRC data and the ASD data

is good, especially considering the spatial resolution differences between the two sensors and the modeling to get surface reflectance data from the Landsat satellite. This measurement establishes an upper limit to the accuracy of the method, given the data available.

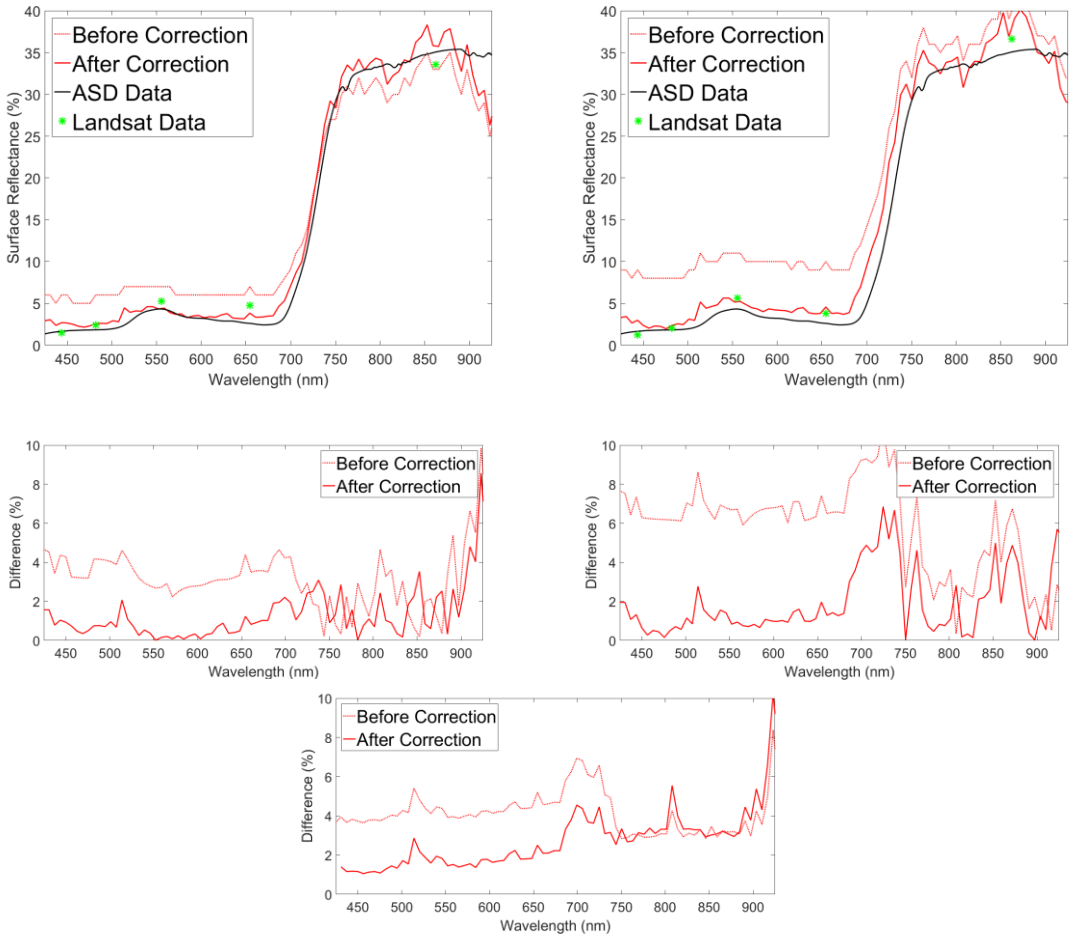


Figure 20: (Upper) Spectra of single pixels of a wheat field before (red dotted) and after (solid red) correction to LaSRC data compared to ASD data (black) and LaSRC data (green asterisks) for the same locations. Before Correction data had the simple atmospheric model applied to compare surface reflectance values directly. (Middle) Difference between hyperspectral data and ASD data before and after correction corresponding to the upper graphs. (Lower) Average difference between hyperspectral data and ASD data before and after correction.

Multi-Year Example

One of the potentially useful results in terms of absolute radiometric correction is illustrated in Figure 21, which shows the difference between the calibrated and uncalibrated data taken one year apart (Appendix B). The hyperspectral data derives its radiometric values from the auto-exposure algorithm of the camera (controlled by the shutter speed) and associated pre-flight laboratory calibrations that do not consider changes in the sensor, lens, illumination conditions or atmosphere. Neglect of these instrumentation and lighting changes can potentially cause the year-to-year vegetation reflectance changes to either be lost or exaggerated. With the addition of the LaSRC data and the processing techniques presented, a more complete story can be seen from year to year. This idea of radiometric calibration, especially to surface reflectance, is of immense importance when trying to draw conclusions based upon temporal data.

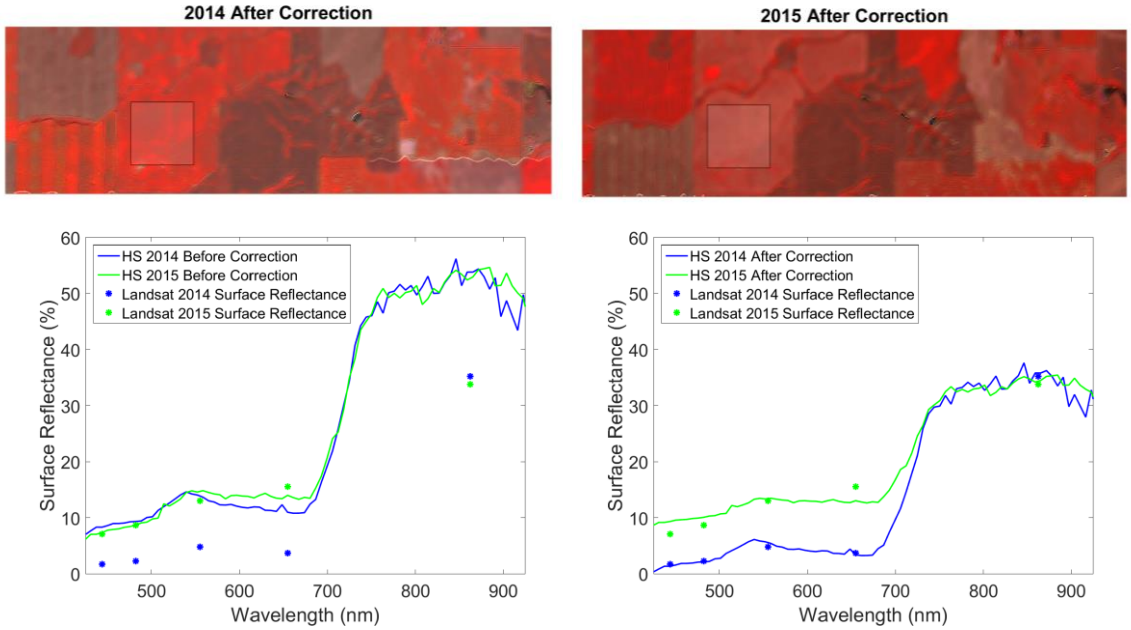


Figure 21: (Upper) False-color IR images of Site C are displayed with the same manual dynamic range adjustment for each image using the average of bands 5–15, 15–25, 65–75 of the hyperspectral data acquired from the same location in two different years (2014 and 2015) after geocorrection and radiometric correction. (Lower) Hyperspectral data of a sample area (box in upper images) plotted before and after correction to the LaSRC data showing the excellent agreement between the data. The sample area was chosen because the land area was uniform in coverage and vegetation type for both years. In the left graph, it is not possible using the hyperspectral data alone to come to any conclusion about changes in the data even though the LaSRC data would suggest changes were present. In the right graph, however the change in the spectra can easily be seen in the hyperspectral data and agrees with the LaSRC data.

While in this work LaSRC data was used as the surface reflectance reference (primarily due to the free access to high quality surface reflectance data), this method can be used with other surface reflectance reference data provided that the necessary spectral response curves are available, the atmospheric correction from radiance to reflectance is sufficient for the problem, and the spatial resolution is adequate.

Conclusion

Field calibration of hyperspectral imagery is difficult, and any method will have advantages and disadvantages. For example, placing a calibration target in the field such as a Spectralon target with a well-defined reflectivity provides a reference spectrum that can be used to correct the flight based hyperspectral imagery. These calibration targets provide excellent calibration near the target, but as one moves away from the target, the calibration becomes less certain. Thus, a grid of targets is needed. Placing targets every 100 m (30 m) would require 1000 (11,000) targets to cover a 10 km^2 area. This number of targets becomes untenable for larger areas, particularly if site access is limited.

The bootstrap method described in this thesis utilizes LaSRC surface reflectance data as a reference target for hyperspectral imagery. This technique relies on several reasonable assumptions. The first set of assumptions is aimed at rectifying the different spatial resolution of the LaSRC data product and the hyperspectral pixel resolution. The first of these two assumptions involves treating the hyperspectral imager as a linear imaging system. This assumption allows a reflectance spectrum to be generated by simply adding the reflectance spectra from several pixels together and dividing by the total number of pixels that are being combined. The second assumption is that the atmosphere changes little over the 30 m pixel resolution associated with the LaSRC surface reflectance data product. The second set of assumptions is aimed at rectifying the difference in the spectral resolution of the LaSRC data product and the hyperspectral imager. The LaSRC data product provides 5 spectral channels in the 425–925 nm spectral region (omitting the Landsat panchromatic band) while the hyperspectral imager provides 80 spectral channels.

This implies that the LaSRC cannot be used to correct each spectral channel of the hyperspectral data independently. However, assuming the responsivity of the hyperspectral imager and the Landsat imager are linear in intensity, the calibration can be completed as described in this thesis. First, the hyperspectral data is combined to produce Landsat equivalent channels using a scaled responsivity, $R_{final} = G * R_{init} + B$, where R_{init} is the initial reflectance, R_{final} is the final reflectance, G is the gain, and B is the bias, and the relative spectral responses of the Landsat channels. Then, G and B are adjusted to minimize the difference between the LaSRC data and the Landsat equivalent channels generated with the hyperspectral data for each pixel. This responsivity is then used to modify the entire hyperspectral spectral reflectance. Thus, the five spectral channels of the LaSRC data are used to generate the two parameters (overall gain and bias) needed to adjust the responsivity of the hyperspectral imager to provide the calibration correction. The advantages of this method include the 30 m grid of reference targets over large areas and minimizes the need for site access. The major disadvantage to this method is the fact that a linear correction is applied to the hyperspectral data which may miss higher order spectral corrections.

Field calibration of flight based hyperspectral imagery is important for producing surface reflectance data over large areas. The most common method of field calibration involves the distribution of calibration targets such as Spectralon (Bruegge et al., 1993; Jackson et al., 1992) to provide a well-defined reference surface reflectance for calibration. However, this method has disadvantages when trying to image large areas with limited site access. In this thesis, a bootstrapping calibration technique was presented that can allow

for calibration of mesoscale hyperspectral imagery. This method is based on three reasonable assumptions described above. Combining calibration methods such as distribution of calibration targets with the bootstrap method presented in this thesis may provide a more robust calibration technique where limited site access and/or limited calibration targets are available.

The bootstrap method of calibration presented in this thesis has been applied to several hyperspectral data sets at various field sites. At an agricultural field site, the calibration technique presented in this thesis was compared to ASD data at seventeen locations. As seen in Figure 20, correcting the hyperspectral data with an atmospheric model results in a 4.16% difference on average as comparted to the ASD data. After applying the calibration correction based on the method presented in this thesis difference was reduced to 2.74% on average. This experimental validation lends credence to the calibration based on the linear responsivity correction based on comparing the Landsat equivalent channels with the LaSRC channels.

The ability to produce hyperspectral imagery over large areas using LaSRC data products as a surface reflectance reference was shown over a test area 5.78 km^2 in extent, but was also done at two nearby areas of 5.86 km^2 and 6.79 km^2 in extent giving a total area of 18.4 km^2 that was calibrated. Furthermore, the hyperspectral data collected allows a temporal comparison of flight-based hyperspectral data.

BIOPHYSICALLY RELEVANT PARAMETERS

Introduction and Motivation

Each hyperspectral pixel contains 80 distinct spectral channels covering the visible and near-IR spectral region (425–925 nm). In this spectral region, vegetation has several features that can be used for classification of land cover, vegetation health, yield prediction, and others. Many of the differences between vegetation types can be subtle so low noise instruments are of paramount importance. One means of noise reduction is to combine adjacent spectral channels, but this comes at a cost of spectral resolution. Another straightforward way to reduce noise is to fit a spline, or some other smoothing fiction, to the data, and in some circumstances this can be of value; however, it does not provide any additional physical insight. Alternative to this type of noise reduction is to re-express the data in a different basis.

One common way of representing data in a different basis is the Fourier transform. There are numerous different bases that may be appropriate to a given problem, whether plane waves, Lagrange functions, spherical harmonics, Bessel functions, and many others. In many of these bases the deviation between the initial signal/function and the representation in the new basis decreases as additional terms are added. In many cases a perfect representation is an infinite sum with each additional term reducing the deviation by a small and smaller amount. A finite series in this case is then an approximation of the original data and since the new basis is made up of smooth functions it is also a means of smoothing the data.

Here is presented a technique that transforms hyperspectral data, which is a collection of distinct spectral bands, into a set of biophysically relevant parameters. The biophysically relevant parameters are the variables in a mathematical model used to represent the 'shape' of the reflectance spectra. This model is chosen to represent features in the spectra that are physically relevant. One benefit of doing a transformation based on fitting the 'shape' of the spectra is that each parameter has input from every spectral band, thus leading to a degree of noise reduction. The cost of such a method is that there is data loss associated with fitting, especially if a poor model is chosen.

The fitting function described below uses a physical interpretation to guide the model components to investigate what is physically happening in the system. The model has been designed to represent vegetation while at the same time having enough degrees of freedom to represent fallow regions. The technique presented here looks at hyperspectral scenes that include vegetation and fallow fields, but can be applied to other hyperspectral imaging scenes by using a different set of basis functions relevant to the physical content of the spectral image.

Fit Functions for Vegetation

The biophysical relevant basis functions used to model the spectral images presented in this work consists of two distinct pieces that are summed to give the final modeled reflectance spectra (Appendix C). The two pieces are referred to as the Red Edge and the Green Peak. In brief, the Red Edge helps to define the baseline reflectance in the visible, the location and behavior of the red edge, and the strength of the near-IR

reflectance. The Green Peak defines the characteristics of the green peak found in the visible region. The model uses a total of nine parameters to fit the reflectance spectra reducing the data associated with the 80 spectral channels recorded for each pixel in the spectral image thus reducing the data by a factor of 8.9. The actual amount of data reduction in terms of raw storage is only 55% because the initial hyperspectral data is represented as int16 numbers (16 bits), or 1280 bits per pixel, and the nine parameters are stored as double-precision floating-point numbers (64 bits), or 576 bits per pixel.

Red Edge

With the success and widespread use of NDVI (Carlson & Ripley, 1997; DeFries & Townshend, 1994; Wójtowicz et al., 2016), the first function that was chosen was the inverse tangent (arctan) function, Figure 22, which serves as a type of step function. A step type function was chosen, because, in a broad sense, vegetation has a low reflectance in the visible and high reflectance in the near-IR. Given the spectral resolution of the hyperspectral camera, it is possible to determine significantly more information than simply a ratio between visible and near-IR as is done with an NDVI measurement. To have enough degrees of freedom to accurately represent the data, this part of the model requires 5 separate variables as shown in equation 4, which defines the red edge function. The first variable gives the baseline in the visible portion of the spectra, labeled $R1$. The second variable measures the difference between the visible baseline and the base level in the near-IR, labeled $R2$. The third variable, arguably the most important, is the location, in wavelength, of the inflection point of the arctan function, labeled $R3$. This value is a measure of the location of the red edge, and can be determined very precisely as compared

to simplistic means of determining the location of the red edge (Carter & Knapp, 2001; Cho & Skidmore, 2006; Dawson & Curran, 1998; Gitelson et al., 1996; Horler et al., 1983; Smith et al., 1997; Zarco-Tejada et al., 2000). The fourth variable deals with how steep the rising edge of the arctan function is in terms of reflectance versus wavelength (Miller et al., 1990), labeled $R4$. The final variable is included to change the curvature of the function near its minimum and maximum to match the curvature more precisely surrounding the red edge, labeled $R5$. While implemented as a Gaussian, this portion of the red edge function could be simplified by performing a Taylor expansion and only keeping the lowest order terms. This would simplify the computation meaning less time would be required to do the fitting. The red edge function, RE, can be written in terms of the hyperspectral wavelengths λ_{HS} as

$$RE(\lambda_{HS}) = R2 * \left[\frac{\tan^{-1}\left((\lambda_{HS} - R3) * R4 * e^{\frac{(\lambda_{HS}-R3)^2}{R5}} \right)}{\pi} + \frac{1}{2} \right] + R1 \quad (4)$$

The red edge function with varying parameters is shown in Figure 22 with the red line having average values and the green line having a decreased curvature near the edges determined by changing parameter R5 (Appendix B). The blue line has steeper slope than the red line determined by parameter R4, and the black line is similar to the red line, but with larger parameters R1 and R2, and a shift in the red edge determined by parameter R3.

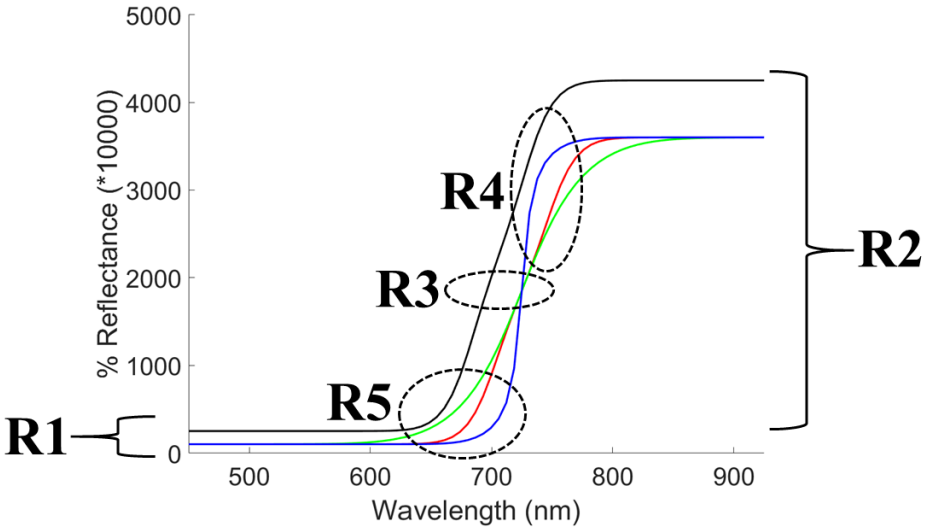


Figure 22: Graphical representation of the different parameters necessary to give the arctan function sufficient freedom to define the red edge well. R1 is the brightness of the pixel in the visible. R2 is similar to NDVI giving the difference between the brightness in the visible and the near-IR. R3 is the location of the inflection point giving the location of the red edge. R4 defines how steep the edge is in terms of reflectance versus wavelength. R5 defines the curvature of the function near the extremes.

Green Peak

The green peak function (GP), the second basis function considers the peak in the visible responsible for the green coloring seen in plants. As is common in natural phenomena, the reflectance peak was first assumed to have a Gaussian distribution as a function of wavelength; however, it was determined experimentally, based on the R^2 or coefficient of determination (Rao, 2009), that a single Gaussian was insufficient to match the behavior seen in the vegetation spectra. R^2 is defined as $R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2}$ where f_is are the predicted values and y_is are data points being fit. Specifically, a single Gaussian left error in the red portion of the spectrum, meaning that the blue side match well, but the red side required a longer 'tail' to model the data. To model the 'tail' of the reflectance

peak that extends towards the near-IR more accurately, an exponentially modified Gaussian (exGaussian) was considered and was found to yield higher R^2 values. An exGaussian is a convolution of a Gaussian and an exponential. The parameters needed to define this function are similar to a Gaussian function with the first parameter related to the area of the peak, labeled $G1$. The second parameter is the location of the peak in wavelength space, labeled $G2$. The third is the width of the peak, labeled $G3$. The fourth and final parameter gives rise to the tail of the exGaussian, which causes a shift in the peak location, labeled $G4$. Thus, the GP function can be written in terms of the hyperspectral wavelengths λ_{HS} as

$$GP(\lambda_{HS}) = G1 * G4 * e^{\frac{(G3*G4)^2}{2} - (\lambda_{HS} - G2)*G4} * normcdf\left(\frac{\lambda_{HS} - G2}{G3} - G3 * G4\right) \quad (5)$$

A normal cumulative distribution function given by $normcdf(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-t^2/2} dt$ is used in this green peak basis function. A graphical representation of what changes in these parameters corresponds to the model reflectance spectra is shown in Figure 23 (Appendix B) with the red line being normal values, the green line having a larger area determined by changing G1 and larger width determined by changing G3, the blue line is shifted by changing G2, and the black line has identical parameters to red line except for a tail because of a larger value of G4.

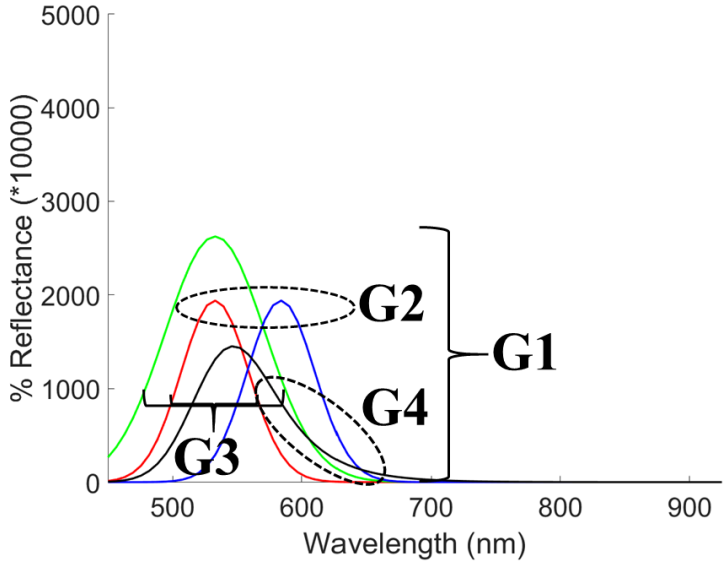


Figure 23: Graphical representation of the different parameters necessary to give the exGaussian sufficient freedom to define the green peak well. G1 defines the area, and therefore the height of the Gaussian. G2 defines the location of the center, and G3 defines the width of the Gaussian peak. G4 defines the exponential that modifies the Gaussian manifesting as an extended tail and an apparent shift in the peak location.

Fitting Examples

The strength of the model can be seen in Figure 24 where the measured and modeled reflectance spectra are shown for pixels containing vegetation (upper left and upper right), sparse vegetation (lower left) and fallow (lower right). The black dots represent the measured reflectance spectra, the red dotted line represents $RE(\lambda_{HS})$, the green dashed line represents the $GP(\lambda_{HS})$, and the magenta line represents the total modeled reflectance spectra $RE(\lambda_{HS}) + GP(\lambda_{HS})$. As the model is designed for vegetation, the R^2 value is greater than 0.99 for vegetative pixels. Reflectance spectra from pixels that are either partially or not vegetated show that the model has enough degrees of freedom to represent these reflectance spectra well, providing excellent fits with R^2 values greater than 0.98.

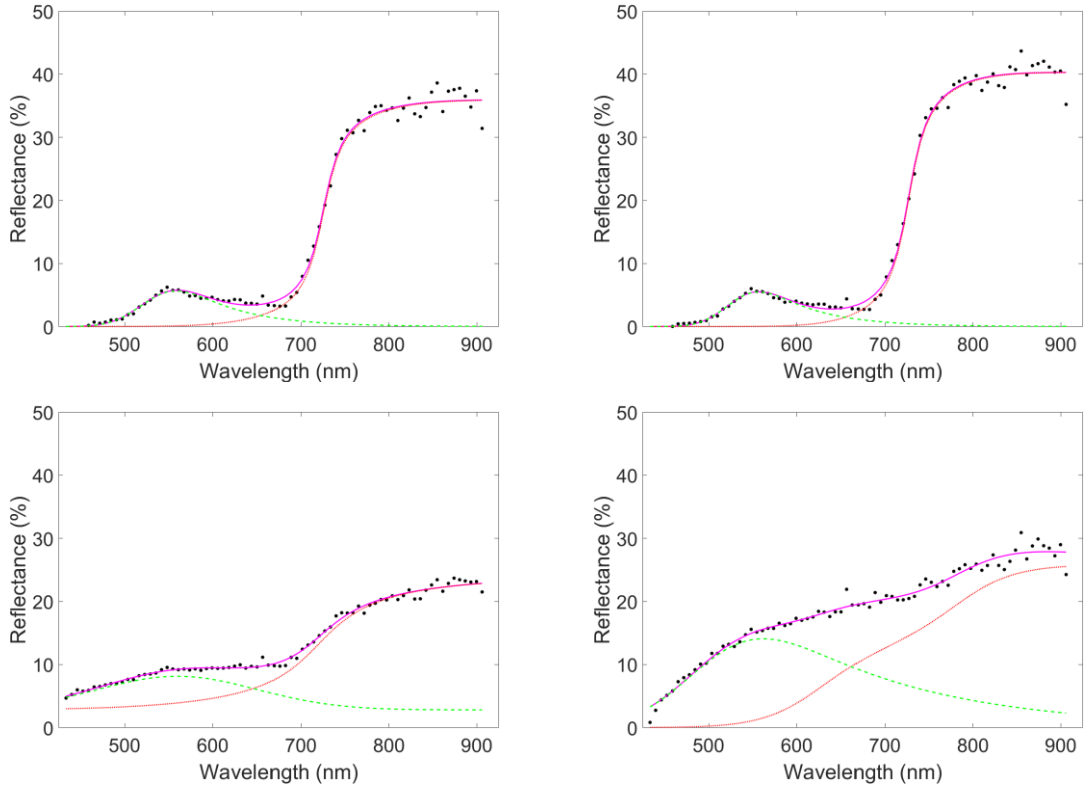


Figure 24: Fits to example pixels containing (Upper Row) vegetation, (Lower Left) sparse vegetation, and (Lower Right) fallow are shown in solid magenta with the original spectra shown as individual black points. Individual portions of the basis are shown: red dotted line shows the red edge function and green dashed line shows the green peak function.

Fitting the reflectance spectra for each individual pixel with this model is a time-consuming process on a standard desktop computer. For example, a quad core 2.7 GHz processor does fitting at a rate of approximately 30 pixels/second or 2.78 pixels/GHz/s (Appendix B). The fitting need only be done once on a dataset before being analyzed in terms of biophysically relevant parameters, is highly parallelizable, and could be done with a field programmable gated array (FPGA) if desired (Kuon et al., 2008). After the fitting is complete, analysis of the entire image can be done in many ways with these physically relevant fit parameters.

Atmospheric Absorption Lines

As evidenced by the examples in Figure 24 the atmospheric absorptions lines corresponding to O_2 and H_2O are absent. This is due to the atmospheric modeling done during the initial processing to turn the at sensor radiance into surface reflectance. If this atmospheric modeling step was not done the data will contain atmospheric absorption lines. This can be taken care of by adding some 4 Gaussian peaks to the fitting model described above. While Voigt functions would be the appropriate lineshape, the spectral resolution is insufficient to differentiate between a Voigt function and a Gaussian and since these peaks are due to atmospheric absorption Doppler broadening will dominate. The two sets of H_2O and the two sets of O_2 peaks can be constrained based on the relative absorptions strength of the lines as well as the relative position between them. The addition of more parameters will increase the time to do the fitting, but can yield additional atmospheric information if desired. This information could be used to determining the H_2O/O_2 column to feed back into an atmospheric model.

Future Direction

Since this work has been focused on vegetation in agricultural areas the basis functions needed to represent vegetation well and have enough freedom to represent fallow fields. This led to the functions described above. However, these basis functions would not work well if the scene of interest was urban in nature where differentiating between several types of non-vegetation may be the goal, or a mining scene involving different

surface minerals. Additionally, as sensors improve and further information becomes available in the UV or Near-IR the basis functions may need to be improved.

As far as inexpensive hyperspectral imagers extending the wavelength range into the near-IR to around 1100 nm is the most likely path of development. If vegetation were still the primary interest this would necessitate allowing for the red edge function to decrease as it extends into the IR, by adding in a negative Gaussian peak, or a piecewise defined linear function.

Urban Example

AVIRIS data, from the Aliso Canyon Line 2 (N34° 8’ 15”, W118° 41’ 19”) flight taken on 01/14/2016, encompassed parts of San Fernando proper and the surrounding hills, parks, and open spaces. This image was cloud free, had 6.4 m pixels (some of the smallest pixels available from AVIRIS), but was not orthorectified so images are not “North-Up.” The simple atmospheric model, mentioned previously, was utilized to turn the data into surface reflectance, but the weakness of the model was evident as the aerosol profile was not rural, and the H_2O column was much thicker than the model. These limitations were mitigated by testing the model in the visible and near-IR regions only and then masking out the regions of large atmospheric absorption (900–1500 nm and 1750–2050 nm) as well as removing the UV portion of the spectra (350–500 nm) that was dominated by scattering (Appendix C). Advanced atmospheric model, such as those mentioned previously, would add a significant amount of useful information to the spectra in these regions, but this analysis was beyond the scope of this thesis.

To include the additional information, two additional exponentially modified Gaussian peaks are added in the short-wave IR (SWIR). These two peaks combined with the red edge and green peaks described previously are used to fit two pixel's spectra as shown in Figure 25. One pixel was chosen to be the center of the Ronald Reagan Freeway near the interchange with the San Diego Freeway. This is a 5 lane (each direction) freeway with no center vegetation median making it an ideal representation of a concrete structure in an urban environment. The other pixel was chosen from a neighborhood on the outskirts of the city. This pixel is most likely a mix of vegetation, asphalt (road), and concrete (sidewalk), but potentially tops of houses/cars as well. This mixture of landscapes would benefit from work being done on spectral unmixing (Boardman, 1993; Boardman et al., 1995; Chen et al., 2017; Keshava & Mustard, 2002; Keshava, 2003; Roberts et al., 2017; Xu & Shi, 2017), but this is beyond the scope of this thesis as well. In both cases, the model represents the data well with an R^2 greater that 0.98 showing the success of the model in representing this urban data. However, this initial work would benefit from a rigorous atmospheric model, as mentioned previously, and further analysis of the urban landscapes to improve the basis functions.

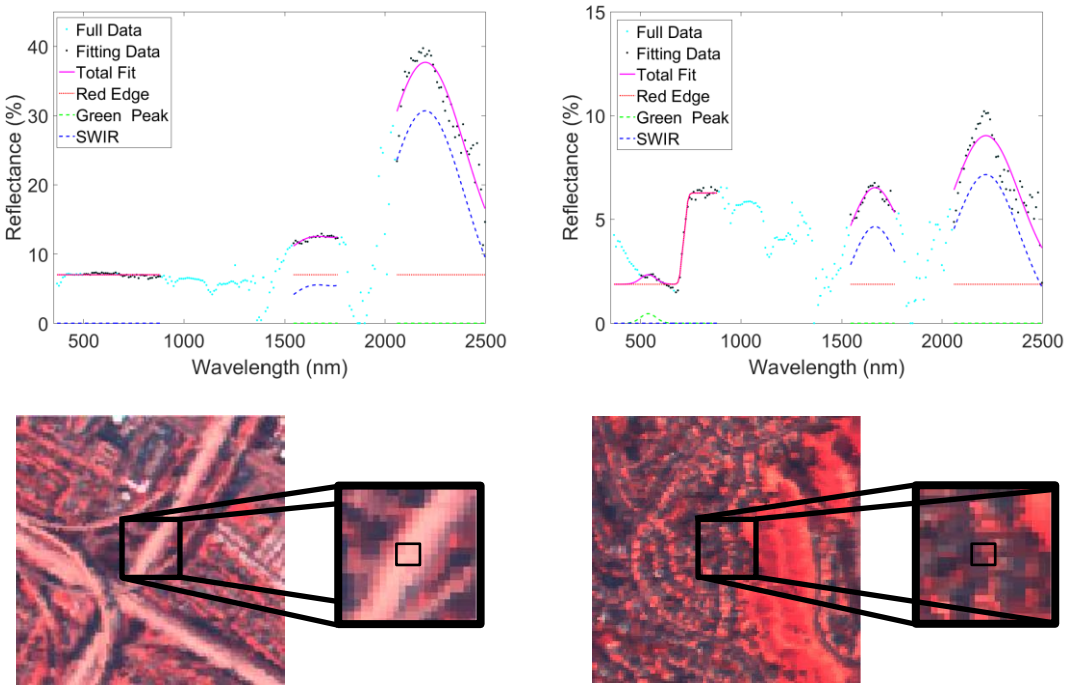


Figure 25: Example urban spectra with fits, note the different vertical (Reflectance) scales for clarity between the interstate (left) and vegetation (right) data in regions with large atmospheric absorption (900–1500 nm and 1750–2050 nm) were removed. Accompanying spatial location, shown in false-color IR of the example pixels (located at the center of each image) is shown below each graph, large area is 646 x 646 m, small area is 134 x 134 m. Green dashed corresponds to the green peak and red dotted is the red edge functions described prevously. Blue dashed corresponds to two added peaks in the short-wave IR (SWIR). These peaks are exponentially modified Gaussians like the green peak function.

Molecular Absorptions

Potentially the most interesting application of this type basis function idea would be to use molecular absorptions as the basis. The reflectance spectrum of a plant is due to physical features at longer wavelengths, but in the visible the spectra is defined by the types/concentrations of various molecular compounds as mentioned in the section on "Vegetation Stress Indicators." If there was prior knowledge of the specific compounds in the vegetation and the relative concentrations expected it would be a simple matter of using

the individual molecular absorptions to fit the spectra. This is provided that the instrument has sufficient signal to noise and spectral resolution to allow differentiation between similar molecules, which may limit this idea to lab based measurements.

Summary

Fitting reflectance spectra as a technique is new and the model presented, while simple, shows the power of such a technique for both vegetated areas as well as urban landscapes. With improved signal-to-noise, finer structural detail can be resolved and with extended spectral range additional structural features become accessible. It is feasible with a more advanced model and improved sensors to glean information about plant structure, health, and composition without having to analyze hundreds of bands at a time.

UNSUPERVISED CLASSIFICATION BASED ON HISTOGRAM SPLITTING

Introduction and Motivation

Despite the similarities present in vegetation reflectance spectra (low in the visible, high in the near-IR, and with a sharp red edge in between), there is variability due to concentration of photosynthetic pigments, structural differences, and on a broad scale there will be a non-homogeneous distribution of other absorbers such as dust and pollen either in the atmosphere or settled on the plants themselves. These effects serve to 'smear out' the reflectance at a given wavelength even in a uniform area. A histogram of an individual band would therefore appear to be a broad peak or a few strongly overlapping peaks. This is one of the issues with many unsupervised classification schemes as even in multi-dimensions these peaks do not form 'islands' that can clearly be distinguished. Many classification schemes, especially simple ones like k-means, rely on some type of Elucidation distance between the center of a cluster and a test pixel to determine how related pixels might be. This has a weakness that clusters will be best represented by n-spheres (circular or 2-sphere in 2D, spherical of 3-sphere in 3D, etc.) in an N-dimensional space. More advanced techniques like ISODATA add a level of complexity in that they merge or split clusters that are either in close proximity or have too large of a deviation in one or more dimension (for example, a hot dog shape may be cut in half to approximate two spheres). Even more advanced techniques exist that attempt to bypass this limitation in many different ways, but regardless of the technique if there are isolated 'islands' classification becomes simpler, if not trivial.

As describe in the chapter on “Biophysically Relevant Parameters” changing bases can change how the data can be analyzed. Using the parameters described (changing the basis of the data) bands that did not exhibit distinct peaks transform into parameters that have in many cases distinct peaks as seen in Figure 26. This figure is data from the entire area of interest and still shows distinctly separate peaks in many of the parameters. The peaks at the high and low ends of the histograms are artifacts of the fitting process due to the choice of bounds, limiting the range of values a parameter could take. The following sections describe the technique and how to determine whether peaks are separable.

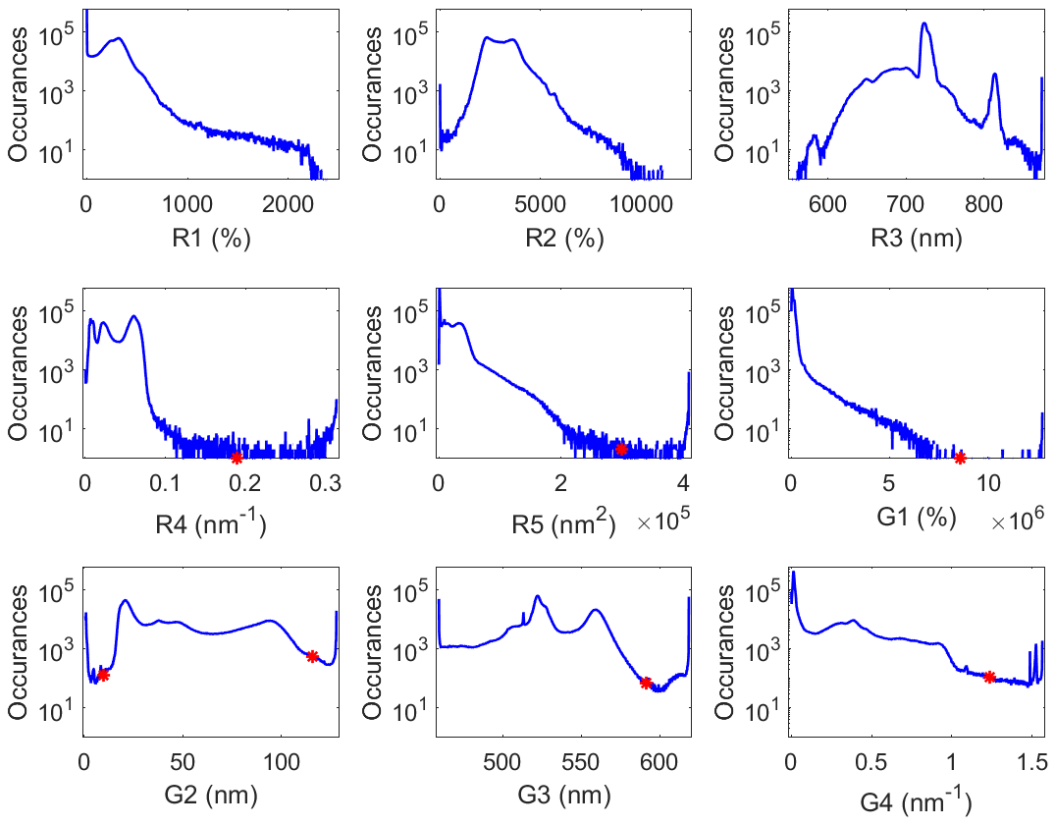


Figure 26: Histograms of parameters found by fitting each individual pixel, vertical axis is occurrences on a log scale. Red asterisks show splitting locations if that parameter was the first to be split. These unsmoothed histograms are for an entire image of Site C during 2016 and become increasingly simplified as splitting continues.

Description of Classification Technique

An unsupervised classification technique was developed based on splitting the histograms associated with the fit parameters used to model the reflectance spectra (Appendix D). A histogram can be generated for each of the nine fit parameters by plotting the number of times a parameter falls within a certain range of values. Certain parameters will exhibit separable peaks that can be split into natural clusters. The unsupervised classification scheme begins by looking for a parameter with separable peaks to do an initial split into clusters. Within each cluster, another parameter with separable peaks is used to split the cluster into further subclusters. This process is repeated multiple times generating a clustered image that can be classified or analyzed as desired.

The first step in applying the unsupervised histogram splitting classification scheme is to provide an initial set of clusters for the spectral image. This can be accomplished by providing a set of broad seed clusters or defining the whole image as a single cluster. Next, one of the fit parameters is chosen, either randomly or simply by starting with the first parameter $R1$, and a histogram is generated for all the pixels within this cluster. For this work, the histogram was generated by taking the range for the parameter of interest P_n, and dividing it by the number of parameter bins, m, so that

$$\Delta P_n = (P_{n,max} - P_{n,\min})/m \tag{6}$$

where $P_{n,max}$ and $P_{n,min}$ are the maximum and minimum values of P_n. The appropriate value for m is determined by the number of data points being examined, here taken to be 5 times the number of data points. The histogram is then a plot of the number of times a parameter value falls within the range $i\Delta P_n$ and $(i+1)\Delta P_n$ where i is an integer. To

minimize the number of extraneous peaks or valleys that may be found in the data, the histogram is smoothed using a loess option of the smooth command in Matlab. The loess option utilizes a local regression using weighted linear least squares and a 2nd degree polynomial model. This choice, while it may change the specific values of the peaks and valleys, preserves their existence and location. The histograms generated for the various parameters plotted on a log scale are shown in Figure 26. From these histograms, peaks and valleys are identified using "findpeaks" in Matlab (Appendix D). In the initial implementation, peaks near the extreme edges, seen in R3, R5, G2, G3, G4 in Figure 26, are considered as being real, though this is not necessarily an accurate representation as the bounds are imposed when fitting the spectra. Future iterations of the technique will require changes to either the bounds when fitting the biophysically relevant parameters or to the routine itself allowing it to ignore peaks near the extreme edges. Changes to the bounds when fitting to the basis functions should give a better representation. Future iterations may also need to enforce a numerical minimum when determining valleys. Splitting locations should only be in sparse areas in the histograms, unlike G2 in Figure 26 where a splitting is found at 550 occurances, though this is rarely an issue after initially splitting such a large are.

With the peaks and valleys in the smoothed histogram found, the next step is to determine where to split the histogram to generate subclusters. For each parameter value corresponding to a peak in the histogram, the higher and lower parameter values where the nearest valleys are located are determined. In a broad sense, these valleys are where the data could be split, but in practice there are three different cases that must be considered to

intelligently split the data as shown in Figure 27. The simplest and most ideal case is where there is a single valley between two adjacent peaks. This directly leads to splitting the data at the location of the valley, Figure 27 (left). The second case is where there are two or more peaks within the same set of valleys. This case only requires not taking into account the redundant peaks, Figure 27 (middle). The final case is the most interesting and most common case, when there is a disagreement between where the split should occur, because there are multiple valleys between the peaks. The data between the valleys must be assigned to one of the adjacent peaks; this is accomplished by determining the center of mass (COM) of the data between the peaks and assigning it to the side it is closest to, Figure 27 (right). The COM is defined to be

$$COM = \frac{\sum \Delta E * x}{\sum \Delta E} \tag{7}$$

where ΔE is the number of elements in the histogram bin centered at x as x takes on all the values of the bin centers between the valleys of interest. While there could be other choices made in this regard, this was the simplest to implement, and is based on the idea that proximity of the parameters corresponds to being physically similar.

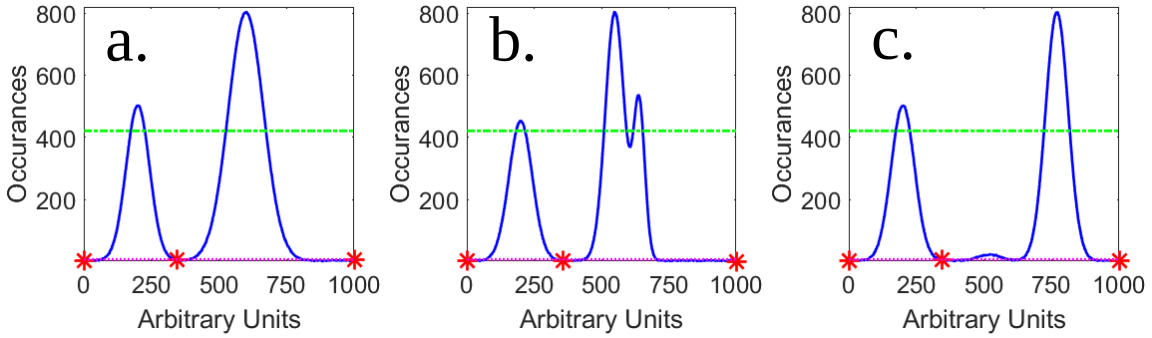


Figure 27: Example data (blue) showing where the histogram splitting technique would split (red asterisks) different types of data. The left plot (a) shows the simplest case with a single peak between each set of valleys. The middle plot (b) shows how multiple peaks are contained between two valleys. Finally, the right plot (c) shows how the center of mass is used to determine that the small middle peaks should be associated with the rightmost peak since it is closer, but is too small to be considered an independent peak. In each plot the dotted magenta line shows the highest value that a valley can be as determined by 20% of the data being below this value, and the green dashed line shows the lowest value a peak can be as determined by 20% of the data being above this value.

With the best locations to split the data determined, the next step is to split the cluster into its subclusters. Example splittings are shown in Figure 26 and refer to where a spitting would occur if that parameter was the first to be examined. Care is taken to keep similar clusters near each other in cluster space. This is accomplished simply by maintaining the numbering scheme when a cluster is split. For example, an image initially has clusters 1–5 and then cluster 2 is split into two subclusters. These new clusters are inserted into the space where the original cluster was located, the new clustering now has clusters 1,(2,3),4,5,6 where (2,3) was previously cluster 2.

This process is repeated for each cluster for the current parameter of interest. Then a new parameter is chosen, again randomly or numerically, and each cluster is tested against this new parameter. This process repeats until all the parameters have been tested multiple times. One of the drawbacks of this approach is that the order in which the

parameters are evaluated can make a difference as to whether a cluster can be split. To this end, the parameters are evaluated multiple times until no further natural splittings are possible, usually this takes 5–10 passes through the 9 parameters. Whether the parameters are evaluated sequentially or randomly does not matter if each parameter is examined enough times such that there are no further possible splits in the data.

Automatic Clustering

Combining clusters automatically can be done in several ways from spectral proximity to spatial location. Here a simple method of spatial proximity is used to combine clusters. One of the features of the histogram splitting technique is that cluster numbers that are close directly correlate to clusters that are similar so there is an additional dimension that can be used to combine clusters. Determining spatial proximity was done by using a co-occurrence matrix with 20 nearest neighbors over the entire range of cluster numbers. The simplest choice in classification space is to only look at the spatial proximity of clusters with a single classification step (i.e. cluster 41 is only compared to cluster 40, and cluster 42). To give equal weight to each potential comparison the co-occurrence matrix is scaled to the cluster size. This can be thought of as a percent co-occurrence so a value of 20 would correspond to every pixel in a class being surrounded by 20 pixels of a second class and a value of 0 would correspond to no pixel being within two pixels of the second class (Appendix D).

Looping through the percent co-occurrence matrix and recombining classes which are above a certain threshold reduces the number of classes, but does not handle small

classes effectively. Looping through the remaining classes with a co-occurrence matrix and combining small clusters based on their spatial proximity only (ignoring the cluster number) and removing clusters below a certain size reduces the number of classes. Due to the nature of the histogram splitting many classes have only 10s of pixels each. This is due to the splitting nature of the technique, and why the histogram technique excels in datasets with large numbers of pixels.

After these two loops though the co-occurrence matrix the histogram splitting technique clusters have been reduced from approximately 200 down to approximately 20. Since ISODATA does not have a classification dimension we cannot apply the first loop directly, instead a measure of contrast is used. Contrast is determined by taking a ratio between the 95% and the 50% in each row of the matrix, this determines how strongly two clusters are related spatially while diminishing the contribution of large randomly distributed classes. This value is again scaled to the size of a cluster to equally weight all clusters. The largest contrast value is determined and those two clusters are merged. After looping through the co-occurrence matrix and combining clusters based on this scaled contrast the number of clusters is reduced to approximately 20 for both the band based and parameter based ISODATA clustering. Classes are then assigned based on maximizing the trace of the confusion/error matrix.

Indian Pines Reference Data and Results

The publicly available Indian Pines data set (Baumgardner et al., 2015) was acquired with the AVIRIS sensor, described previously, over the Indian Pines test site in

Northwestern Indiana. This data set was 145 x 145 pixels in extent and originally contained 224 spectral reflectance bands in the wavelength range 400–2500 nm, which was reduced to 200 bands after removal of bands covering water absorption. The Indian Pines dataset is designated into sixteen classes two-thirds agriculture land and one-third forest or other natural perennial vegetation and includes two major dual lane highways, a rail line, low density housing, other built structures, and smaller roads. The scene was acquired in June, so some of the crops present such as corn and soybeans are in early stages of growth with less than 5% coverage.

Performance Measures

To evaluate the performance of the histogram method versus ISODATA a classification matrix was first determined based on location of ground reference data to clusters contained within the area. Since the number of clusters found is not constrained the matrix will be $n * c$ where n is the number of reference data classes and c is the number of clusters found, this can be reduced to an $n * (n + 1)$ matrix, the last column representing unclassified areas. A perfect method would produce an $n * n$ matrix with non-zero values only along the diagonal. A typical entry q_{ij} shows how many samples belonging to class i that have been assigned to class j.

Performance measures evaluated are: individual class accuracy (PA_i) or producer's accuracy, reliability (UA_j) or user accuracy, average accuracy ($\overline{PA}$), average reliability ($\overline{UA}$), overall accuracy (A_{tot}) and Cohen's kappa (κ) (Cohen, 1960; Senthilnath et al., 2013). These measures are defined to be:

$$PA_i = \frac{q_{ii}}{\sum_{j=1}^{n} q_{ij}} \quad (8)$$

$$UA_j = \frac{q_{jj}}{\sum_{i=1}^{n+1} q_{ij}} \quad (9)$$

$$\overline{PA} = \frac{1}{n}\sum_{i=1}^{n} PA_i \quad (10)$$

$$\overline{UA} = \frac{1}{n}\sum_{j=1}^{n} UA_j \quad (11)$$

$$A_{tot} = \frac{1}{N}\sum_{i=1}^{n} q_{ii} \quad (12)$$

Where q_{ii} are correctly classified samples, q_{ij} are incorrectly classified samples, n is the number of classes, and $N = \sum_i \sum_j q_{ij}$ is the number of samples in the entire dataset. The entire dataset of pixels with reference data was used for performance measures.

Cohen's kappa coefficient is a statistic which measures inter-rater agreement for categorical items and is a robust measure compared to simple percent agreement calculation, since it considers the agreement occurring purely by chance.

Results and Discussion

As a means of validating the unsupervised classification technique based on the histogram splitting method, data was clustered by the both the histogram method and ISODATA using the default settings in ERDAS Imagine (Number of Classes = 1 to 200, Minimum Size=0.01%, Minimum Distance=4, Maximum SD=5, Max Merges=1, Maximum Iterations=50, Convergence Threshold=0.99). ISODATA was run on both the 53 spectral bands and the parameters from the fit model described in this thesis.

A comparison of clusters before and after merging is shown in Figure 28 for the three different classification methods. Qualitatively the differences between using

parameters versus bands for ISODATA gives very different results, but after merging begin to converge on clusters. Several fields are clustered similarly between the three methods potentially pointing to variability within the fields that is not expressed in the reference data.

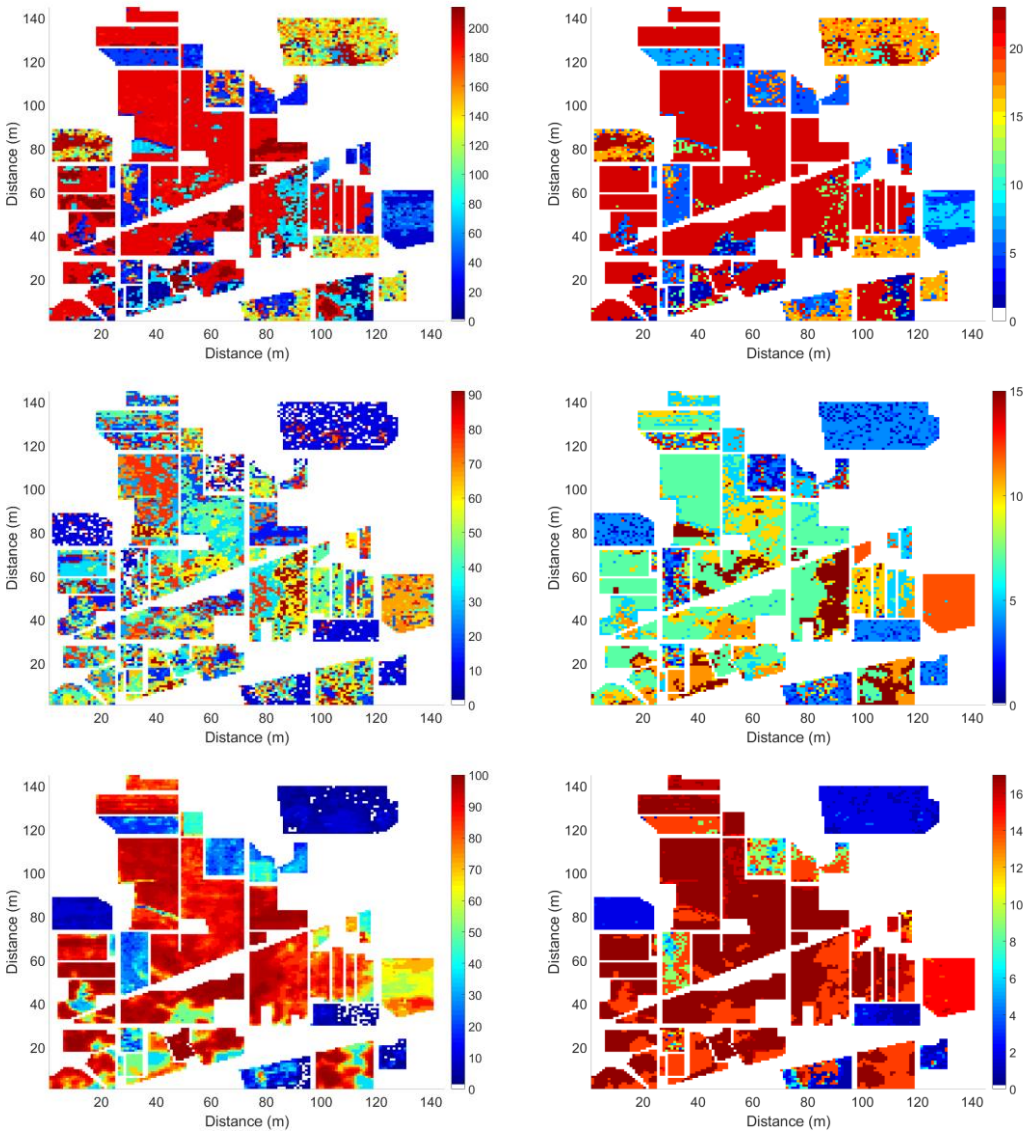


Figure 28: (Upper) Indian Pines classified using histogram splitting, (Middle) ISODATA on the fit parameters, and (Lower) ISODATA on the spectral bands; (Left) before and (Right) after automatically combining clusters are shown with a mask corresponding to available reference data.

The resulting data is quantitatively shown in Table 2 showing individual class accuracy (PA_i) or producer's accuracy, reliability (UA_i) or user accuracy, average accuracy ($\overline{PA}$), average reliability ($\overline{UA}$), overall accuracy (A_{tot}) and Cohen's kappa (κ).

Table 2: Classification efficacy for Indian Pines Data set comparing the histogram splitting method, ISODATA on biophysical fit parameters, ISODATA on hyperspectral bands both before and after automatic cluster merging. NaN values of the User Accuracy is due to the respective algorithm not classifying any pixels in that class.

Class type	**Sample Size**	**Unmerged Histogram Splitting PA/UA%**	**Merged Histogram Splitting PA/UA%**	**Unmerged ISODATA Parameters PA/UA%**	**Merged ISODATA Parameters PA/UA%**	**Unmerged ISODATA Bands PA/UA%**	**Merged ISODATA Bands PA/UA%**
A1: Alfalfa	46	15.2 / 77.8	0.00 / NaN	50.0 / 100.	0.00 / NaN	41.3 / 100.	0.00 / NaN
A2: Corn-notill	1428	12.5 / 41.5	15.1 / 38.4	15.6 / 55.1	22.3 / 57.6	15.8 / 50.2	0.07 / 100.
A3: Corn-mintill	830	2.29 / 95.0	1.08 / 69.2	10.2 / 58.6	11.3 / 44.6	16.0 / 84.7	7.59 / 30.1
A4: Corn	237	1.69 / 66.7	0.84 / 100.	22.8 / 87.1	2.53 / 100.	18.6 / 100.	0.00 / NaN
A5: Grass-pasture	483	9.32 / 52.3	1.45 / 87.5	5.59 / 100.	5.59 / 100.	31.9 / 99.4	3.52 / 73.9
A6: Grass-trees	730	55.1 / 42.9	62.1 / 32.2	20.6 / 60.2	20.1 / 60.2	11.5 / 97.7	30.8 / 91.5
A7: Grass-pasture-cut	28	3.57 / 100.	0.00 / NaN	64.3 / 100.	3.57 / 100.	60.7 / 100.	0.00 / NaN
A8: Hay-windrowed	478	32.0 / 79.3	41.6 / 51.4	30.8 / 99.3	49.2 / 90.7	20.3 / 99.0	96.7 / 88.9
A9: Oats	20	20.0 / 100.	0.00 / NaN	20.0 / 100.	20.0 / 100.	25.0 / 100.	0.00 / NaN
A10: Soybean-notill	972	16.7 / 61.8	21.7 / 63.9	15.4 / 78.5	37.0 / 65.5	14.9 / 78.0	42.5 / 24.8
A11: Soybean-mintill	2455	68.4 / 42.1	77.4 / 42.4	16.1 / 46.0	47.7 / 36.9	20.3 / 55.1	74.2 / 37.8
A12: Soybean-clean	593	7.25 / 48.3	4.72 / 65.1	9.44 / 78.9	26.1 / 52.4	7.59 / 100.	0.00 / NaN
A13: Wheat	205	15.6 / 54.2	6.83 / 56.0	31.7 / 84.4	31.7 / 84.4	41.5 / 89.4	0.00 / NaN
A14: Woods	1265	60.7 / 58.6	90.8 / 61.0	52.2 / 76.3	74.0 / 59.5	24.0 / 75.8	72.2 / 71.5
A15: Buildings-Grass-Trees-Drives	386	2.07 / 100.	0.78 / 100.	17.9 / 75.0	19.4 / 71.4	5.96 / 100.	26.9 / 96.3
A16: Stone-Steel-Towers	93	3.23 / 100.	1.08 / 100.	38.7 / 100.	0.00 / NaN	68.8 / 65.3	0.00 / NaN
	$\overline{PA}/\overline{UA}$	20.4 / 66.4	20.3 / 61.9	26.3 / 76.4	23.2 / 73.1	26.5 / 87.2	22.2 / 68.3
STATISTICS	A_{tot}	**34.3**	**40.9**	**21.1**	**35.1**	**19.0**	**39.2**
	$\boldsymbol{\kappa}$	**24.8**	**30.5**	**17.6**	**26.9**	**15.8**	**28.5**

The histogram splitting method was shown to be an improvement over ISODATA on the reference Indian Pines data before any type of clustering in terms of Cohen's kappa and overall accuracy. After automated clustering/merging both ISODATA and the histogram method were improved, with significant improvement to the ISODATA technique in terms of overall accuracy and Cohen's kappa. Average accuracy suffered as many small classes were not identified at all. Small classes such as "Alfalfa" and "Grass-pasture-cut" are identified better before merging while medium to large classes, above approximately 400 pixels, are improved due to the nature of merging small clusters into larger clusters which is why average accuracy is reduced after merging.

Extensions to the histogram splitting technique would be to examine mixed dimensions as there may be splittings that do not occur along the parameter axes, or shaping N-Dimensional surfaces around clusters in parameter space. Also, further work on the automated clustering algorithm could be done to examine both parameter space and physical proximity at the same time, this could benefit any clustering technique as the simple implementation here showed.

Examples from Test Areas

Since reference data is not available for the test areas it is only possible to qualitatively assign descriptions to clusters. However, given the limited knowledge of the areas it is possible to assign some meaning to the clusters and evaluate qualitatively how the method is working. One example classification, shown in Figure 29 (upper), is for

Area 1, the southernmost region that surrounds the CO_2 production well. A false-color image of the area is included in Figure 29 (lower) for comparison.

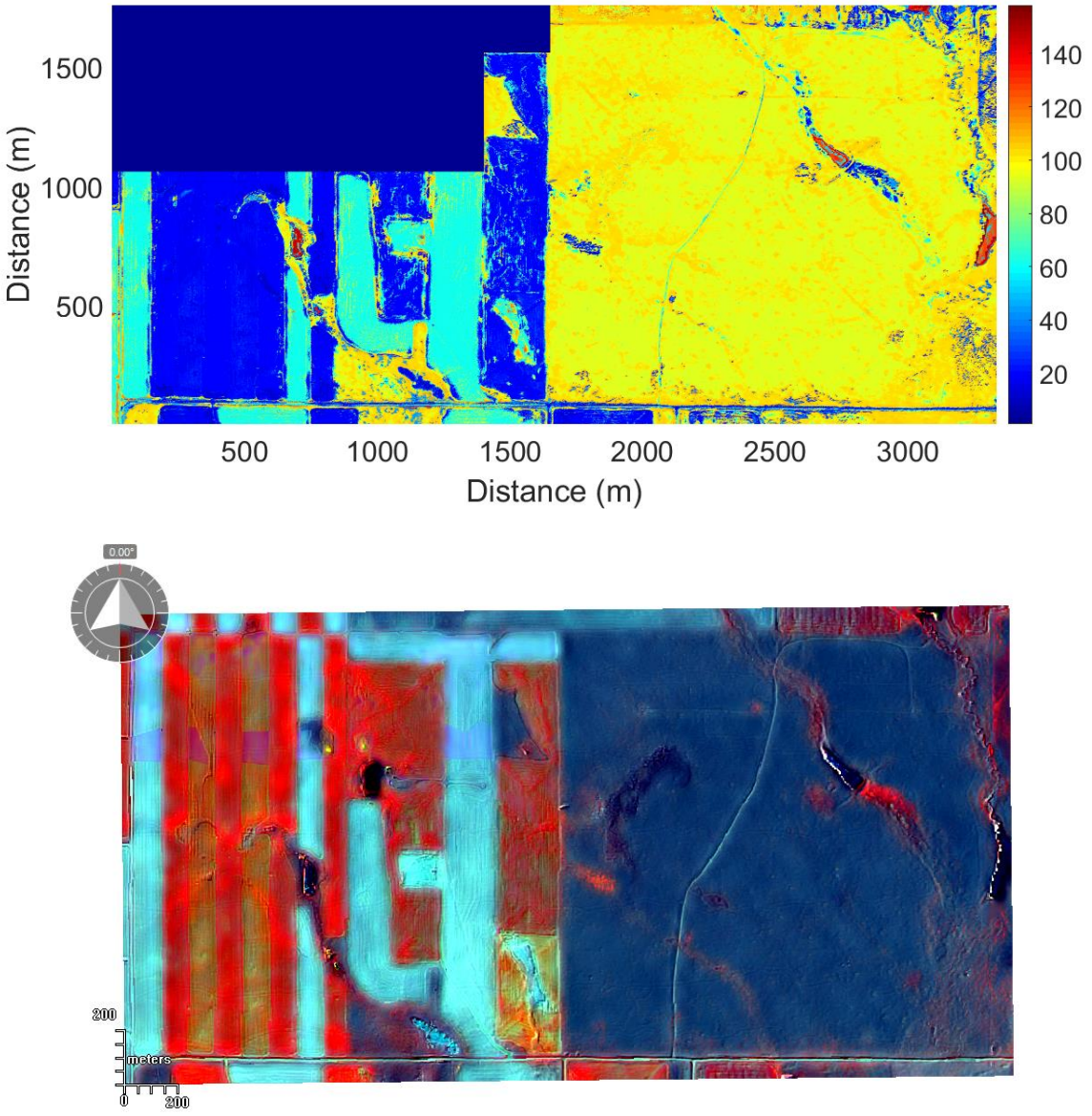


Figure 29: (Upper) Clustering of Area 1. Upper left corner is omitted due to overexposure problems in the camera. Light blue are fallow fields, dark blue is wheat fields, yellow is grassland, reds are water as determined from analysis of the (Lower) data, displayed with a standard deviation stretch in false-color IR using bands 10, 20, 70 of the hyperspectral camera which correspond to 490.8, 554.7, 874.2 nm.

This area during 2014 had a field in the southwest that was planted with alternating rows of fallow/wheat. The middle section of fields was mostly fallow with a few areas planted with wheat. The right half is state land and consists primarily of native grasses with a single road through and two ponds it in the northeast with green vegetation surrounding. All of which can be seen in the clustering.

This clustering generated 156 clusters, but as seen in Figure 30, most of the clusters contain less than a faction of a percent of the total area. Since the area is large, 580 ha, this means that these small clusters can still be approximately 1 ha. There are dominant clusters that make up a substantial portion of the image could be the usual behavior of an area with the smaller clusters being anomalous in comparison. This could be reduced to approximately 30 clusters using the automated clustering technique described previously.

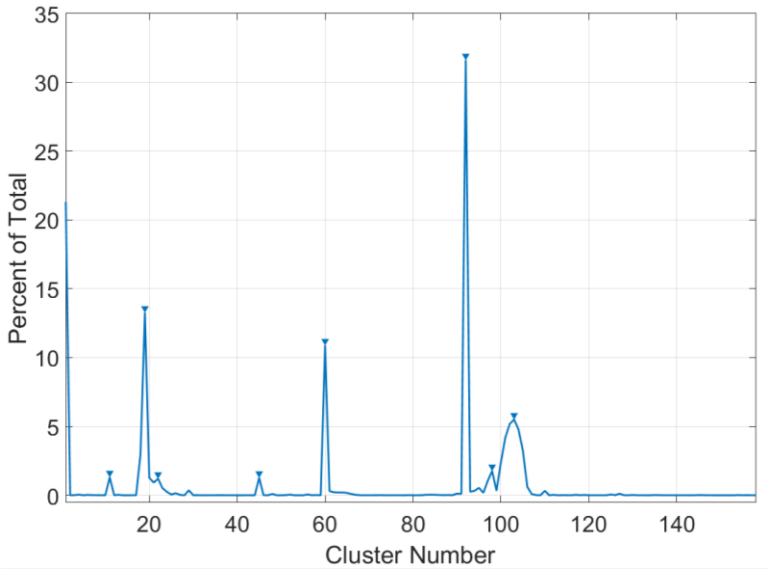


Figure 30: Size plot for each of the 156 clusters determined using the unsupervised classification technique. A sizeable percentage of the area is contained within just a dozen classes.

Without reference data from the area quantitative analysis is not possible for these areas, but the general behavior of the clusters can be used to determine anomalies, or to look at changes in land usage as detailed in the following chapter.

Summary

The unsupervised histogram splitting technique presented in this paper is shown to be an improvement over the standard unsupervised ISODATA clustering technique with an overall accuracy of 34.3/19.0% before merging and 40.9/39.2% after merging. This improvement is also seen as an improvement of kappa before/after merging of 24.8/30.5 for the histogram splitting technique compared to 15.8/28.5 for ISODATA. Without reference data, quantitative conclusions could not be drawn for the test areas, but qualitatively the clusters could be associated with known land use.

The unsupervised classification technique using natural splittings of the histograms of biophysically relevant parameters is one tool developed to work with the 'big data' associated with hyperspectral data. This technique was developed to work with the biophysically relevant fit parameters as there were identifiable natural splittings; however, the idea can be extended into other areas. Not being based on Euclidian distance allows for this technique to naturally represent the clusters and could be extended to work with other types of data that exhibit natural splittings in some basis.

MULTI-YEAR DATA

Introduction and Motivation

Multi-temporal imaging is a valuable tool for monitoring a multitude of changes over time from land coverage/usage change (Byrne et al., 1980; Lunetta et al., 2006; Rogan et al., 2002) to disease spread (Franke & Menz, 2007; Liangyun et al., 2004; Liu et al., 2006) to urban expansion (Gomez-Chova et al., 2006; Li & Yeh, 1988; Maktav & Erbek, 2005; Xian et al., 2008) and many others (Demirel et al., 2011; Hong et al., 2010; Travelletti et al., 2012; Tripathy et al., 1996). This technique has been driven by free access to high quality satellite data and the long-time frame of operation of these satellites, most prominently Landsat and MODIS. Multi-spectral satellites have dominated the multi-temporal field, but multi-temporal hyperspectral data would allow for further advances in the areas mentioned previously.

While there are several multi-temporal hyperspectral studies utilizing ground based sampling (Lausch et al., 2013; Nguyen & Lee, 2006; Strachan et al., 2002; Stuckens et al., 2011; Xie et al., 2013), this thesis focuses on flight-based multi-temporal hyperspectral studies (Franke & Menz, 2007; Liu et al., 2010) of which there are limited examples, especially at the mesoscale. With constantly improving technology in terms of the sensors and continually improvements in atmospheric models there is an ever-increasing array of problems that can be addressed by multi-temporal hyperspectral imaging. A significant percentage of these problems will require high quality radiometrically referenced data to draw quantitative conclusions. As it stands many multi-temporal studies rely on minimally

processed satellite or flight data that usually lacks any type of absolute calibration to surface reflectance (Goenaga et al., 2013; Petitjean et al., 2012; Yuan et al., 2015). This approach relies on unchanging atmospheric conditions and that the calibration of the sensor is maintained, limiting the quantitative information that can be obtained.

One specific use of multi-temporal hyperspectral data is monitoring of geologic carbon sequestration sites and airborne remote sensing is one technique proposed for monitoring the large areas associated with carbon sequestration sites. Flight based hyperspectral imaging has the potential to monitor vegetation for signs of stress as described in previous chapters. Remote sensing-based surveys of carbon sequestration sites by flight based hyperspectral imaging can then be used to direct more expensive, time consuming, and resource intensive sensors (such as hand held CO_2 sensors) to potential problem areas.

Initial experiments demonstrating the ability of hyperspectral imaging to detect stressed vegetation were conducted during controlled sub-surface release experiments at the Zero Emission Research Technology (ZERT) field site. Keith et al. (Keith et al., 2009) demonstrated the ability to detect vegetation stress using a ground based hyperspectral instrument. Subsequent work by Bellante et al. and others (Bellante et al., 2013; Male et al., 2010; Pickles et al., 1999; Spangler et al., 2010), also performed at the ZERT field site, demonstrated the ability of a flight based hyperspectral imaging system to detect the evolution of the vegetation stress resulting from a sub-surface release. During this experiment, eight flights were conducted over the ZERT field sites, and data was collected

over an area of approximately 1 ha. Georeferencing corrections were achieved using ground based targets that could have been used for atmospheric correction as well.

Initial demonstrations of hyperspectral imaging indicate that it is a viable method of monitoring carbon sequestration sites. Work presented in this thesis has demonstrated the ability to use U.S. Geological Survey (USGS) 0.3 m resolution orthoimages for georectification and the Landsat 8 surface reflectance data product to produce radiometrically referenced large area hyperspectral images with minimal ground access. Furthermore, a method of fitting reflectance spectra with basis functions based on biophysically relevant fit parameters as a means of data and noise reduction has been demonstrated. Additionally, using these fit parameters, an unsupervised classification technique based on histogram splitting of the fit parameters has been demonstrated. These georectification, surface reflectance referencing, and unsupervised classification techniques based on biologically relevant fit parameters are applied to three hyperspectral imaging flights conducted on 06/21/2014, 06/24/2015, and 06/26/2016 at the Big Sky Carbon Sequestration Partnership (BSCSP) site in north-central Montana. Results from this unsupervised classification technique are used to look at changes in the surface reflectance over the time series. Even though no CO_2 leak was present, controlled or otherwise, it was possible to detect local anomalies within a single data set as well as to compare multiple data sets to remove persistent anomalies and detect new anomalies. These anomalies serve as a proxy for detection of a CO_2 leak. If CO_2 was present in the soil in higher concentrations than the surrounding area the vegetation would appear different and would be detected as a local anomaly.

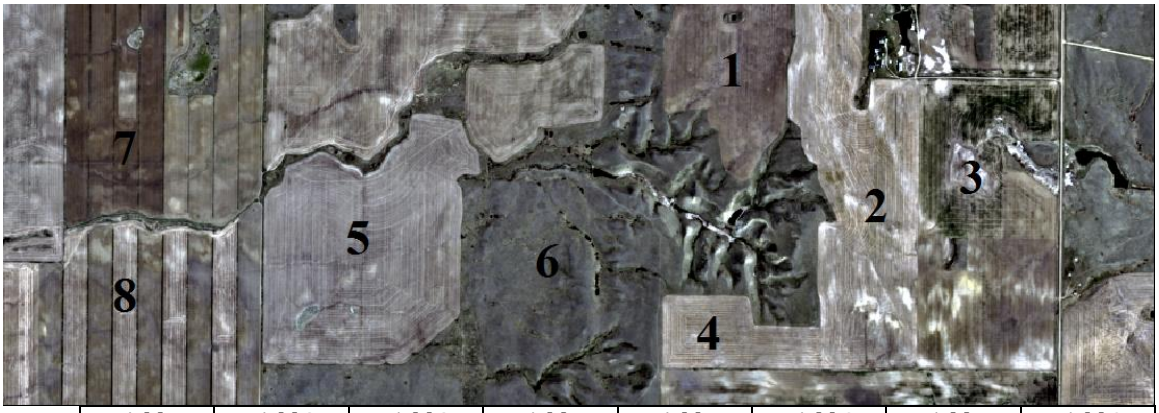
Materials and Methods

The study area and imaging system were described previously in Chapter 1, and the processing was described in Chapter 2. While the biophysically relevant parameters and the unsupervised classification technique were described in Chapters 3 and 4 respectively a brief summery will be given as it specifically relates to Site C.

Field Reference Data

As with many restricted-access sites information about specific land use was limited. The self-reported land use provided by landowners is detailed in Figure 31. Limited site access was obtained in 2015 for fields 2 and 3. This was the only site access during the study years and consisted of visual observations and photographs only.

Additionally, historical rainfall data was obtained for the area from the National Resources Conservation Services (NRCS) for Shelby, MT (Station 7500, 48° 30’ N, 111° 51’ W, elevation 1003 m) located 55 km from Site C. No other stations that monitor rainfall and had data for this period of time were near the site.



	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6	Field 7	Field 8
2014	Fallow	Barley	Barley	Fallow	Barley	Grassland	Fallow*	NR
2015	Spring Wheat	**W**: Fallow **E**: Barley	Barley	Spring Wheat	Barley	Grassland	NR	Fallow*
2016	Fallow	Barley	Barley	Fallow	Field Peas	Grassland	NR	NR

*Determined based on visual examination, not used for analysis.
NR is Not Reported by landowner, but containing some type of vegetation, not used for analysis.

Figure 31: Map of Site C with field labels and corresponding table of land use as reported by the land owners for the 3 years under investigation.

Classification

Classification was performed using an unsupervised histogram splitting classification method based on biophysically based parameters as described previously on all three data sets. In this method, each individual pixel is fit using a set of biophysically relevant basis functions having 9 parameters associated with it. The reduction from 80 spectral bands to 9 parameters provides data reduction as well as noise reduction, but also re-expresses the data in a physically meaningful way. The biophysically relevant basis functions used consist of two distinct functions, referred to as the red edge and the green peak, that are summed to give the final modeled reflectance spectra. The equations for the red edge basis function is

$$RE(\lambda_{HS}) = R2 * \left[\frac{\tan^{-1}\left((\lambda_{HS} - R3) * R4 * e^{\frac{(\lambda_{HS}-R3)^2}{R5}} \right)}{\pi} + \frac{1}{2} \right] + R1 \quad (13)$$

and defines the baseline reflectance in the visible, the location and behavior of the red edge, and the strength of the near-IR reflectance. The green peak basis function is

$$GP(\lambda_{HS}) = G1 * G4 * e^{\frac{(G3*G4)^2}{2} - (\lambda_{HS}-G2)*G4} * normcdf\left(\frac{\lambda_{HS}-G2}{G3} - G3 * G4\right) \quad (14)$$

and defines the characteristics of the green peak found in the visible region. The parameters are described briefly in Table 3 in terms of their meaning/effects.

Table 3: Description of parameters associated with biophysically relevant basis functions used in the fitting of the reflectance spectra of individual pixels.

Name	Description
R1	Baseline in the visible portion of the spectra
R2	Difference between the visible baseline (R1) and the level of the near-IR
R3	Location, in wavelength, of the inflection point of the arctan function, and therefore the location of the red edge
R4	Steepness of the edge of the arctan function is in terms of reflectance versus wavelength
R5	Changes the curvature of the arctan symmetrically near its minimum and maximum
G1	Related to the total area of the green peak
G2	Location of the green peak in wavelength
G3	Width of the green peak
G4	Exponential modifier that gives rise to the 'tail' of the otherwise Gaussian peak and causes a shift towards the red in the apparent green peak location

Each dataset has 9 parameters associated with each pixel, so the classification was done on the full 27 parameters (9 parameters per year for each of the 3 yearly datasets). During the classification, a parameter is chosen and a histogram is generated for that parameter. Certain parameters will exhibit peaks in the histogram that can be easily

separated forming into natural clusters. The unsupervised classification scheme looks for a parameter with separable peaks to do an initial split into clusters. Once this is done, within each cluster parameters are examined until another parameter with separable peaks is found. This new parameter is used to split the cluster into further subclusters. This process is repeated multiple times generating a clustered image that does not possess any further natural splittings and can be classified or analyzed as desired.

Specifically, this classification on the full set of 27 parameters and resulted in less than 10 dominant clusters and approximately 100 minor clusters. Dominant clusters are clusters that contain greater than 4% of the total area, minor clusters are the clusters that contain less than 4% of the total area. This cutoff was chosen based on visual inspection of the sizes of individual clusters to do a qualitative comparison of a limited number of clusters. Dominant clusters were analyzed for their behavior over the 3 data sets.

Single Data Set Anomaly Detection

Within a single data set it is possible to map local anomalies based on how different the parameters for a given pixel are from the surrounding pixels. The measure of anomalousness was based on the number of Median Absolute Deviations (MADs) for each parameter for a given pixel area surrounding the center pixel being examined (Appendix E). MAD is defined as the median of the absolute deviations from the median, or $MAD = median(\,|X_i - median(X)|\,)$, and is a robust measure of the variability of a sample of quantitative univariate data (Howell, 2014; Leys et al., 2013). The amount of anomalousness for a pixel is defined as the number of MADs from the median, or $Anomaly = \frac{|median(X) - Pixel|}{MAD}$. This is similar to the number of standard deviations from

the mean. The total anomalousness for each pixel is the sum across all 9 parameters. Determining the spatial extent for the area in question was a compromise between including large areas, thereby better representing the behavior of the local area, the processing time required, and the relative change when moving to larger pixel area. For this work, the percent change between different sized pixel areas was minimized at 41x41 pixels, as shown in the upper right graph of Figure 32. While the most appropriate pixel area will depend on the spatial resolution of the data and the particular ecological system being studied, 41x41 pixels will be used for this work. This choice of pixel area is further justified by the plots in Figure 32. The upper left graph of Figure 32 shows a histogram of the total MAD across all 9 parameters versus the percentage of occurrences for different sized areas ranging from 11x11 to 61x61 pixels. As the pixel area increases the change in the histograms decreases relative to the next smallest size and the peaks shift towards larger MADs. This diminishing return with larger areas coupled with the upper right graph showing an exponential increase in processing time as the area increases and the relative change between different sizes being minimized at 41x41 pixels leads to the choice of using the 41x41 pixel size to represent the local area. Qualitatively this choice can be seen by looking at an area believed to be the boundary to a region that only has water a portion of the year. This area is shown in the plots in the middle and lower rows of Figure 32. Anomalous areas were visually well distinguished using the 41x41 pixel area as compared to smaller areas, and as the area increased the difference in appearance compared to the next smallest size was minimal.

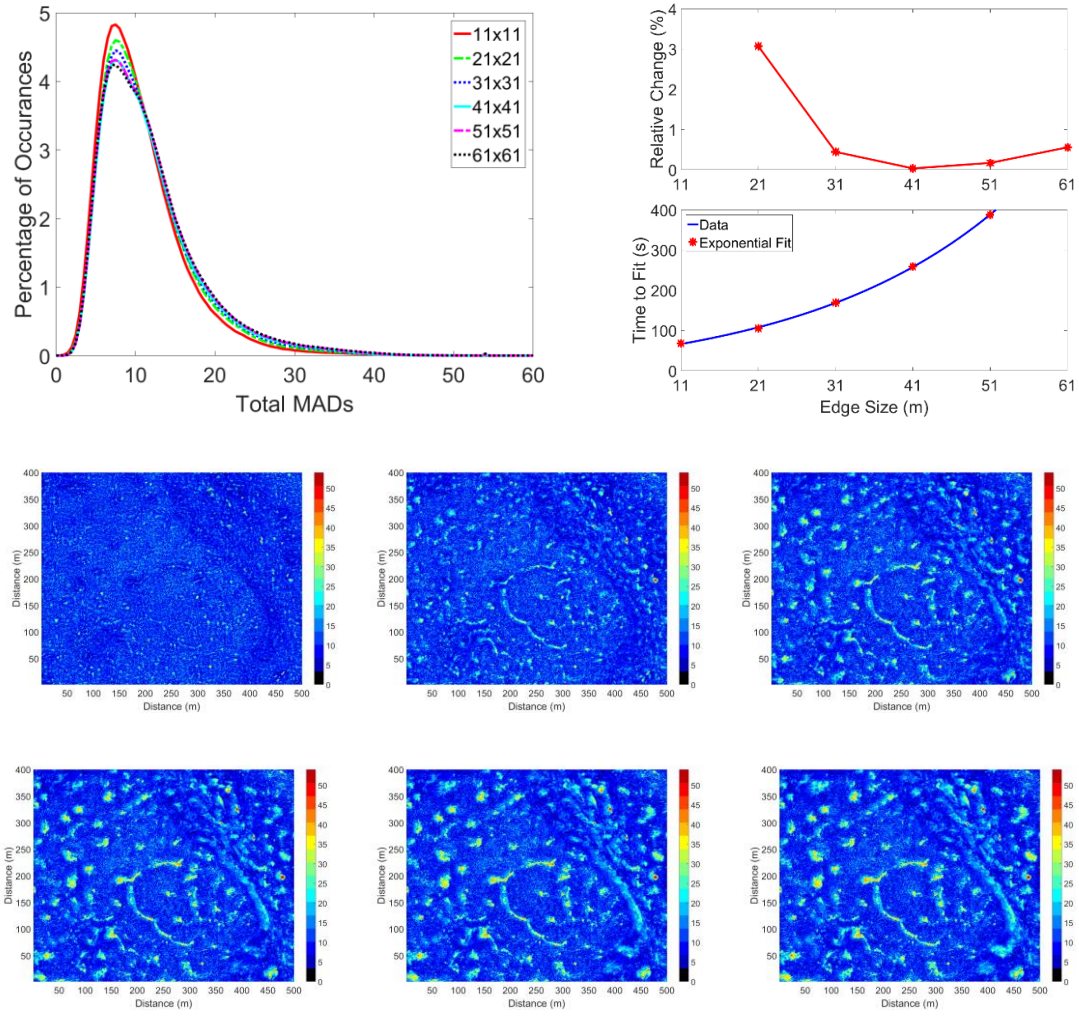


Figure 32: (Upper Left) Histogram of the total MAD across all 9 parameters at different sizes areas. (Upper Right) Relative Change between different Edge Sizes showing the small change at 41x41 and Time (based on fitting a 501x501 pixel area on a 3x3GHz processor) versus Edge Size showing the exponential increase with larger sizes. (Middle and Lower Rows) Example area at different sizes areas: 11x11, 21x21, 31x31, 41x41, 51x51, and 61x61. Z-axis is total MADs; larger numbers are increasingly anomalous in relation to surrounding area.

The deviations, as noted by the high MADs (colors away from blues towards yellow then red on the color bar), from the surrounding may be due to various levels of stress within a plant species, diverse species within the same area, different land use, and other

possibilities as well. A CO_2 leak would manifest itself as a plant stress initially, especially in agricultural land that are uniformly planted with a single plant species.

Multi-Data Set Anomaly Detection

With radiometrically referenced (or radiometrically calibrated) data it is possible to track changes based on having knowledge of the end members in an area (Somers & Asner, 2012; Somers & Asner, 2013). This type of analysis only required radiometrically consistent data as it is not necessary to maintain the same end member library for each set of data. This could be due to imperfect sensor calibration, but more likely would be affected by the natural variation in the reflectance spectra during various stages in its life cycle. This variation can be considered by determining the end members based on the changes over the course of the entire data set (Hemissi et al., 2013).

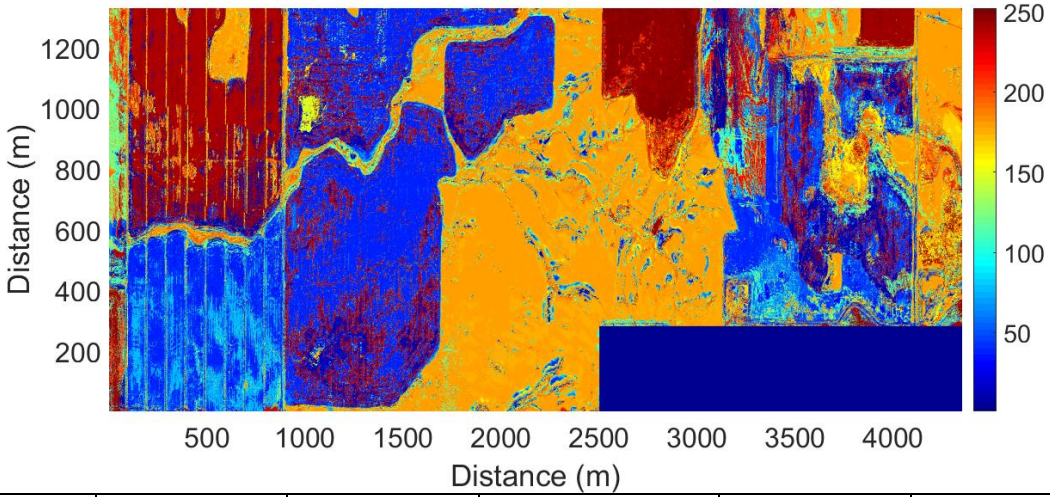
It is also quite common to utilize invariant sites for calibration (Gevaert et al., 2015; Kerekes et al., 2006; Ong & Cudahy, 2002; Zarco-Tejada et al., 2005). This method works well if the invariant sites are distributed throughout the area and the illumination does not change during data collection (such as form a passing cloud). To ensure invariant sites, access is required at multiple times (ideally before each flight) to ensure that the sites are truly invariant and not covered with dust, vegetation, or some other contaminant that could change the spectra. Pseudo-invariant targets remove much of this difficulty if available and properly spatially distributed (Hadjimitsis et al., 2009).

The extension of looking at a single data set to multiple data sets can be as simple as subtracting one map from another. Assuming the spatial registration between the images is smaller than the spatial scale of expected size of anomalies then the difference in the

total MADs between the data sets will be a measure of the increase/decrease of how anomalous a pixel is in relation to the surrounding area. Large increases in the MAD point to an area that has increased vegetation stress relative to the surrounding area, which might be related to either a CO_2 leak or some other localized environmental change. This is just one simple means of utilizing the temporal component of the hyperspectral data.

Dominant Cluster Analysis

The histogram splitting method returned a cluster map of 250 clusters as seen in Figure 33. With the limited field reference data available, it is possible to broadly assign behaviors to the clusters as shown in the table in Figure 33. These behaviors are indicative of the broad scale land use and could be used to determine management areas.



Year	Dark Red	Light Red	Yellow/Orange	Light Blue	Dark Blue
2014	Fallow	Fallow	Rangeland	Planted	Planted
2015	Planted	Planted	Rangeland	Fallow	Planted
2016	Fallow	Planted	Rangeland	Planted	Planted

Figure 33: Histogram based classification using 3 years of biophysical parameters as inputs for a total of 27 parameters. Table shows how color corresponds with land use over the three years.

Depending on the scientific question under investigation it may be necessary to merge clusters either manually, using an automated algorithm, or by defining management areas. However, here the choice has been made to look at the dominant clusters for the large-scale behavior. This assumes that the dominant clusters are representative of the large-scale behavior of the landscape. This assumption is based on Figure 34 that shows the relative percentage of the total area of each cluster. There are 6 clusters that encompass 64% of the study area. These clusters are shown in Figure 35. These dominant clusters could also be used as training data for a supervised classification method as well.

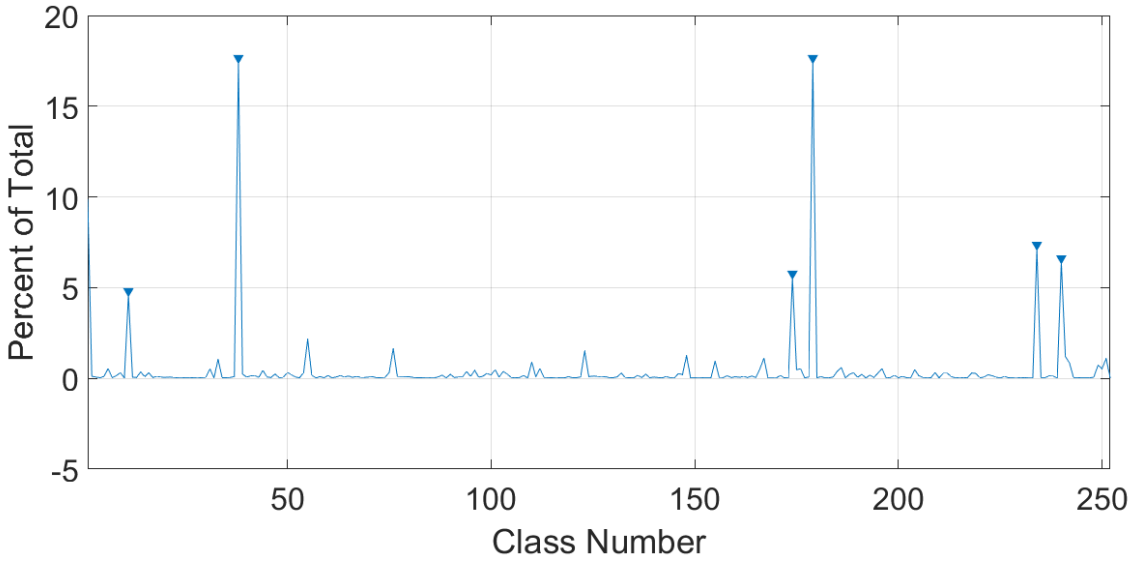


Figure 34: Relative percentage of individual clusters, here 6 clusters make up 64% of the study area, and if numerically similar classes are included 68% of the study area is included.

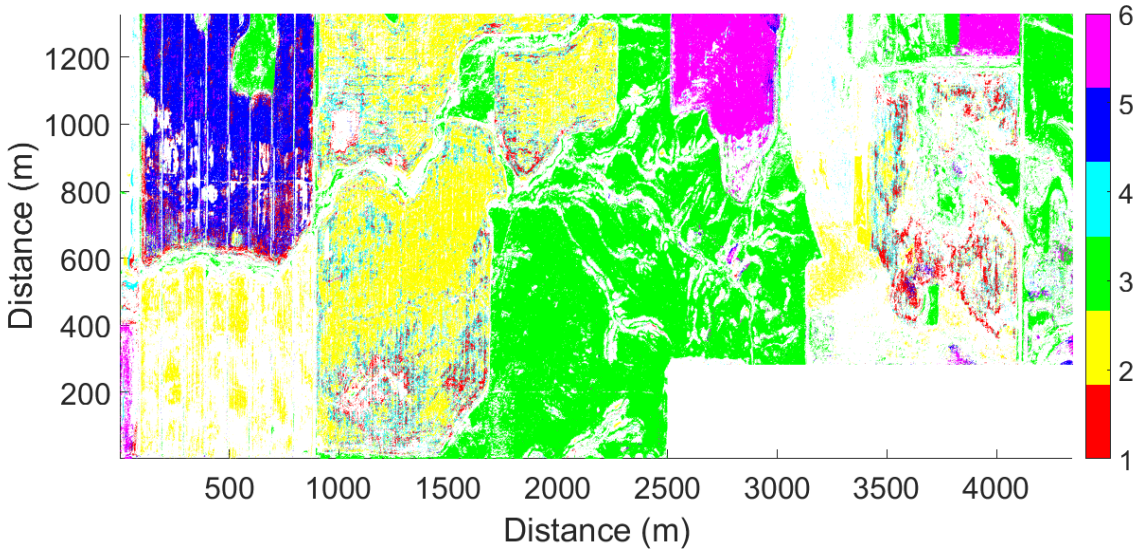


Figure 35: Dominant clusters. Most of the clusters are constrained to field boundaries and appear to be representative of the large-scale behavior of areas noted in Figure 33.

Examination of each dominant cluster gives the spectral plots in Figure 36. The plots are based on the biophysical parameters and appear smoother than the 80 spectral band data. Examples of the 80 spectral band data, are included in the lower plots of Figure 36 for comparison. Cluster 2 is located primarily in the field labeled 5 in Figure 31. This field was reported to contain Barley in 2014, Barley again in 2015, and Yellow Field Peas in 2016. If environmental conditions were the same it would be expected that the spectra from 2014 (solid lines) and 2015 (dotted lines) would be very similar, but this is not the case. The spectra in 2015 is higher in the visible, and slightly lower in the near-IR. This type of change is a common indicator of plant stress (Carter, et al., 1992; Carter and Young, 1993; Carter and Knapp, 2001; Chapin, 1991). Upon examination of the historical rainfall for this area there is evidence that 2015 was a dry year as compared to 2014, as shown in Figure 37. This lack of moisture could be the cause of this stress. However, the grassland, green plot in Figure 36, do not show this relative stress between 2014 and 2015, and in fact

are very uniform through all three years. Since the field was not allowed a fallow year between plantings this relative stress could be due to depleted soil nutrients, or it could simply be due to grasslands being suited for the environment and being less affected by the dry year.

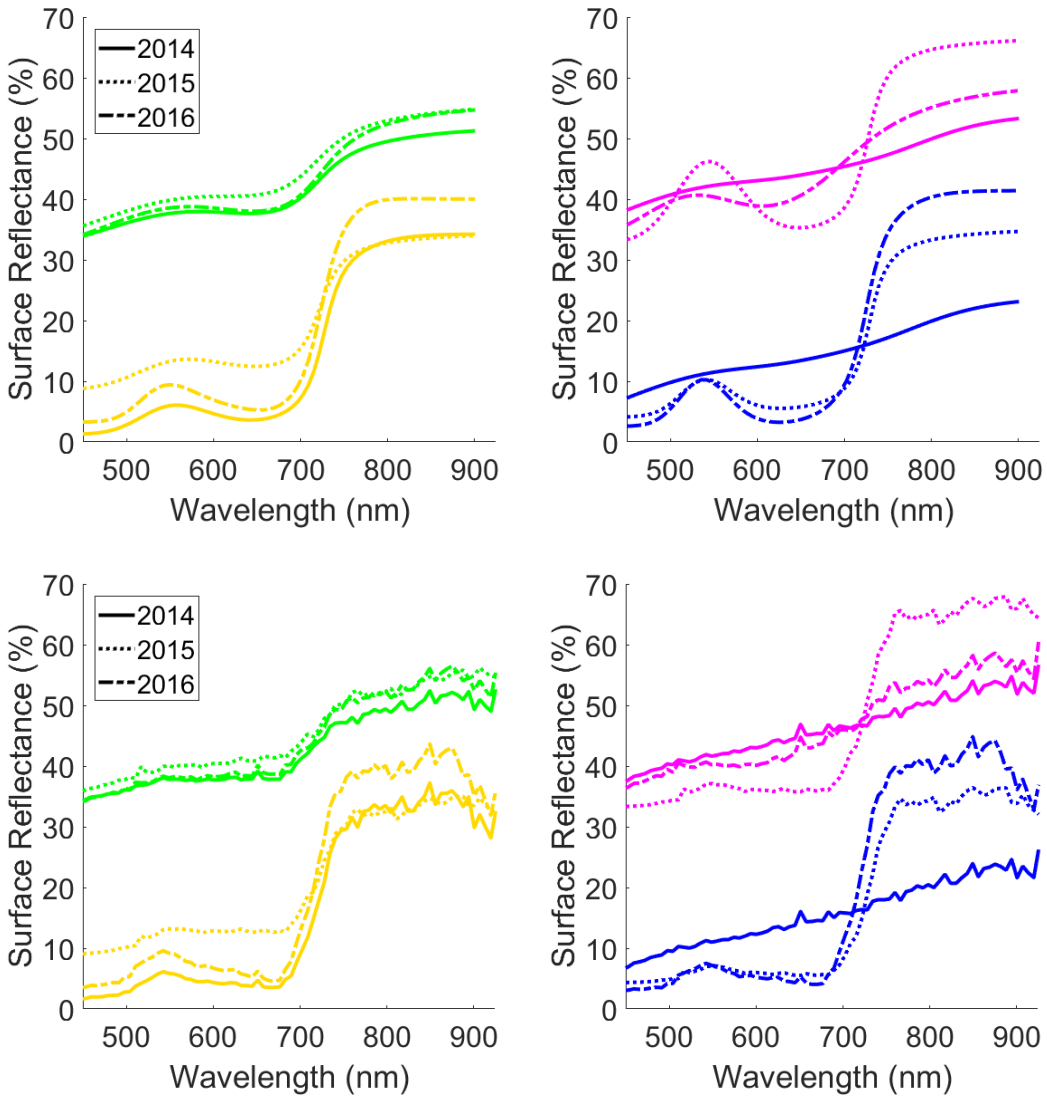


Figure 36: Mean reflectance spectra of parameters (Upper) and spectral bands (Lower) for selected dominant clusters based on biophysical parameters, colors correspond to the same areas in Figure 35. Data is offset by 30% for clusters 3 and 6 for clarity.

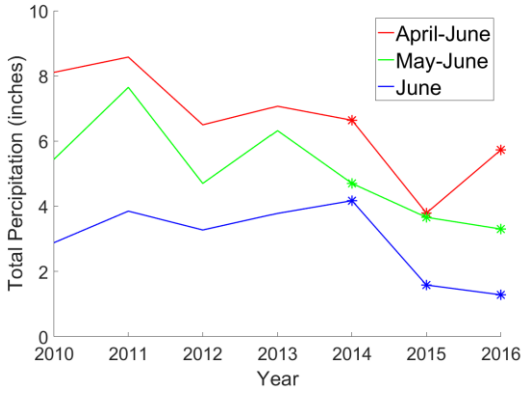


Figure 37: Historical Monthly Rainfall for Shelby, MT obtained from the NRCS. Precipitation data marked with asterisk are when hyperspectral data was obtained at the end of June. 2015 had markedly less rainfall than 2014.

Anomaly Detection

Single Data Set Anomaly Map

A map of suspected anomalies for a single data set, taken in 2015, is shown in Figure 38. Gullies are observed as anomalies in the otherwise uniform grassland as they contain vegetation that is greener than the surrounding native grasses as well as water, as seen in Figure 39 (Left). Also, the rows between the fallow planting rows appear as anomalies, due to residual vegetation that was not tilled when the field was left fallow for the year, as qualitatively identified in the hyperspectral data. The final area to point out is on the east edge, Field 3, where barley has been planted contained a saline seep that made growing conditions less than ideal in the surrounding area. Field 3 had large weedy areas throughout, Figure 39 (Middle), and stunted vegetation, Figure 39 (Right), which were observed as anomalies in Figure 38.

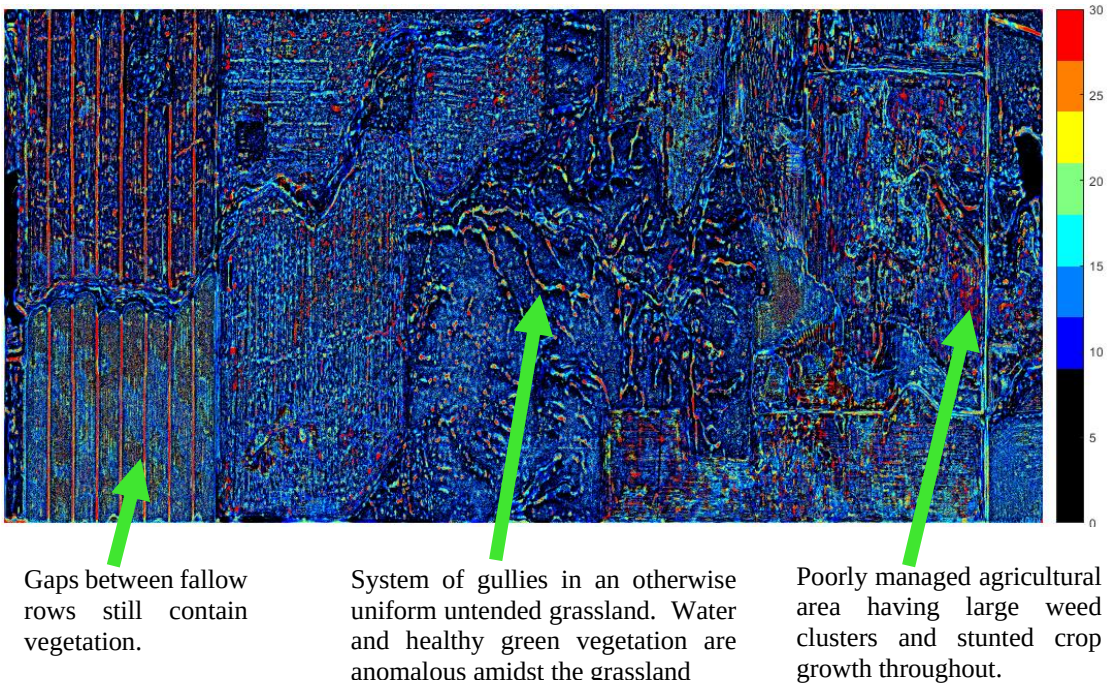


Figure 38: Anomaly map for data taken in 2015. Vertical color scale corresponds to the number of median absolute deviations a pixel is from the median of the surrounding 41x41 pixel region, summed over all 9 biophysical parameters. Larger numbers are more likely to be anomalies.

Figure 39: Example photographs of areas seen as anomalies in Figure 38. (Left) Gullies and surrounding grassland showing greener vegetation near the water at the bottom of the gullies. (Middle) An example patch of weeds located in Field 3. (Right) Stunted crops and sporadic vegetation throughout Field 3.

There are three areas that are worth noting specifically in this anomaly map. The first is a location of an oil well (API #25101072270000) (Wells Search, n.d.) with the surrounding soil being barren shown in Figure 40 (Left). The second area is located near the oil well and is where when the field was being seeded the seeder stopped before the edge either due to running out of seed or from a slight GPS misalignment. This area seen in Figure 40 (Left) is of the same scale as the local area under consideration (41x41 pixels) and shows up as a barbell shape. The third area, mentioned previously, consisted of numerous weed patches in a barley field (southern portion of Field 2) and a nearby saline seep shown in Figure 40 (Right) qualitatively identified during the limited site access to this area.

The oil well is expected to be a persistent anomaly that will appear in every data set. The nearby area that was missed when seeding should be an anomaly that is only present for a single growing season. The size of the saline seep may change depending on the rainfall for the season though the location will be fixed. Finally, the weeded areas are expected to vary in size from year to year if the field is not sprayed, and if sprayed there may be other patches in different areas.

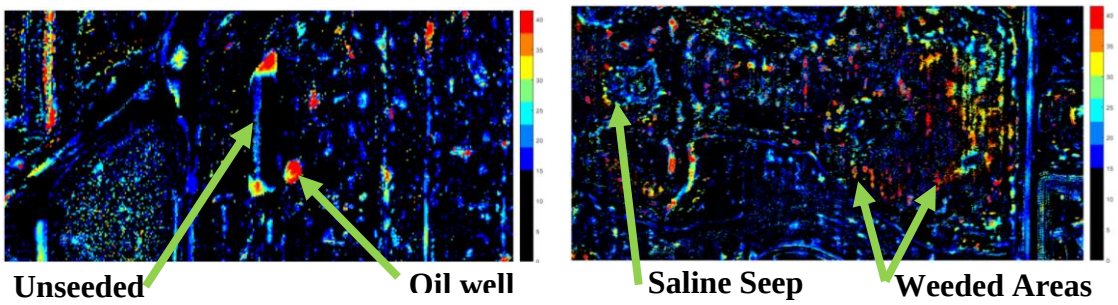


Figure 40: Examples of identifiable anomalous areas in 2015 data. Red/orange areas are more anomalous than blue/black areas. Left: An oil well causing barren soil around it and an unseeded region are both anomalous in an otherwise uniformly planted barley field. Right: A barley field with a saline seep and numerous weed patches throughout.

While a single data set can detect local anomalies, some that can be positively identified, a single data set has limited utility because that many of the anomalies will be present in multiple data sets. However, with radiometrically referenced data it is possible to examine multiple data sets to remove these persistent local anomalies and focus solely on new local anomalies that may result from the CO_2 leak. With enough temporal resolution, easily obtainable with this type of flight-based system, it would be possible to focus solely on temporal anomalies, but since this work consisted of data separated by a year the focus was on spatial anomalies.

Multi-Data Set Anomaly Map

One of the most basic approaches in comparing data sets is to take the difference between the individual anomaly maps. This generates a map that shows how much a given area has changed relative to its surrounding area. This is particularly useful for agricultural areas where the land cover changes dramatically from one year to the next or between plantings during a single growing season. Areas that are more anomalous, either because

they are new or because they have become more different from their local environment, can be considered as new anomalies. A small subset of this type of map is shown in Figure 41 where a cutoff of 10 MADs is imposed for visualization of the anomalies (Appendix B). This means that areas in red are greater than 10 MADs different in 2016 than in 2015. The opposite is true of regions in green, they were more different in 2015 than in 2016. Finally, black are regions are not different enough between the data sets to appear as anomalies.

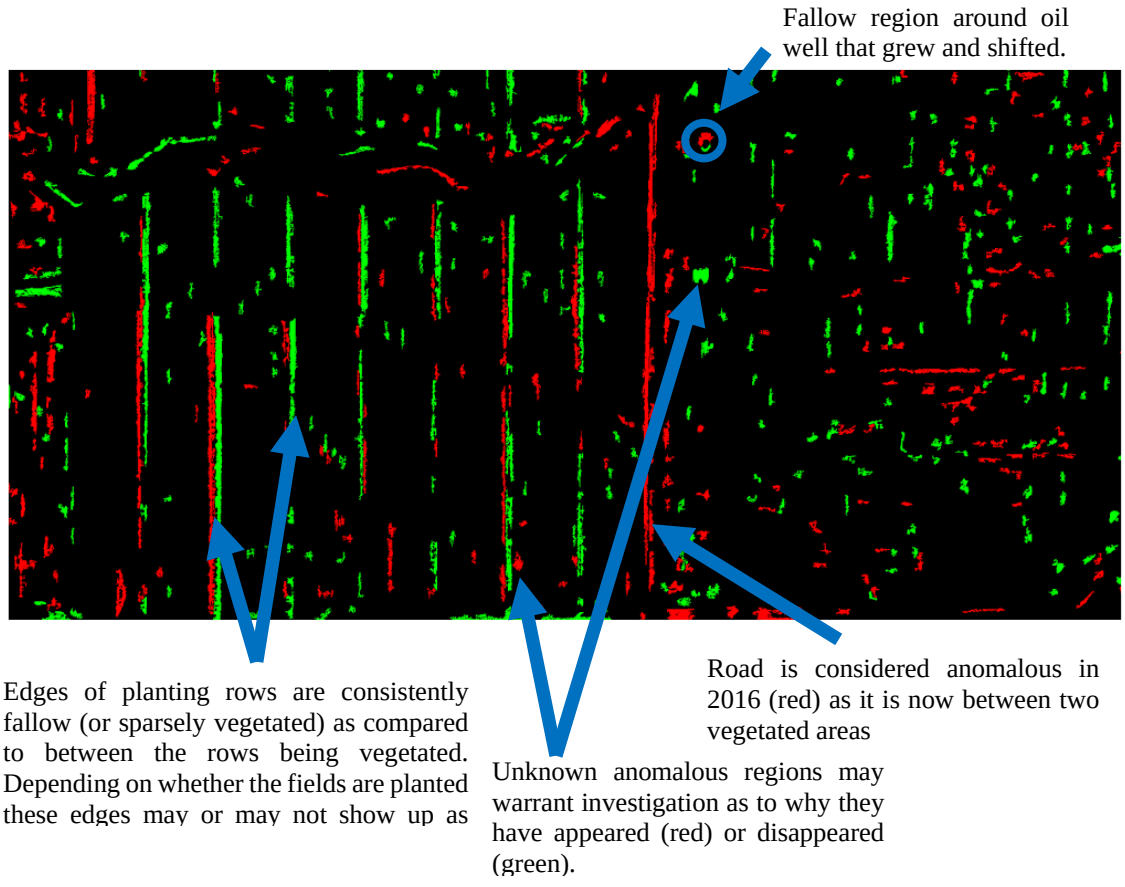


Figure 41: Anomaly change map from 2014 and 2015. Persistent features whose local environment changes, such as the road, appear as anomalies. There are areas that do not have simple explanations that may warrant further scrutiny such as the anomaly at the top of the image.

There are some weaknesses to this approach such as the road that shows up as an anomaly when it runs between two planted fields, but not when only one of the fields is planted. This type of anomaly would be improved as further data are analyzed as there would be additional comparisons between different land use combinations. Similarly, anomalies between planting rows (or fallow rows) in the field in the lower left are seen due to the farmer consistently planting in the same rows. Edges of these rows are consistently fallow (or sparsely vegetated) as compared to between the rows being vegetated so depending on whether the fields are planted these edges may or may not show up as anomalous.

Looking specifically at the positively identified anomalies described previously, namely the oil well, the unseeded region, the saline seep, and the numerous weed patches, as test cases for the technique several conclusions can be drawn. On initial examination, the location of the oil well appears to have grown moved as evidenced by the green and red crescent shapes in Figure 41. If the area had grown or shrank there would be a single colored circular shape, and if were purely movement (such as imperfect spatial registration) the crescents would be the same size. This apparent change in size can be attributed to the fact that in 2015 the barren area was compared to a barley field that had a young crop present so many of the individual pixels still contained a strong signal from the soil that would make the area less different than the surrounding as compared to 2016 when it is compared to a lush pea crop.

The unseeded region near the oil well is seen to disappear (shown as green in Figure 41) except for a small area in the center that appeared in the 2016 data (shown in red). This area can also be identified as a much smaller unseeded area. This appearance is again related to comparing a fallow area against a lush green crop; however, it also shows a weakness to simply examining the difference between data sets.

An alternative means of displaying and working with the data is to define an anomaly cutoff for each data set individually. The working definition was that any pixel with a total MAD greater than 20 was considered anomalous (this corresponds to twice the mean MAD across the entire image). In Figure 42, red areas are anomalies only seen in 2015 as seen earlier, green areas are anomalies in the 2016 data set, and blue are areas that were anomalous in both years. By displaying the data this way it is easier to see how the spatial extent of the oil well appears to have changed, but also it shows the anomalies due to unseeded regions. Displaying the data in in this way also shows how persistent anomalies may be subtly changing and hints at the effects of spatial registration.

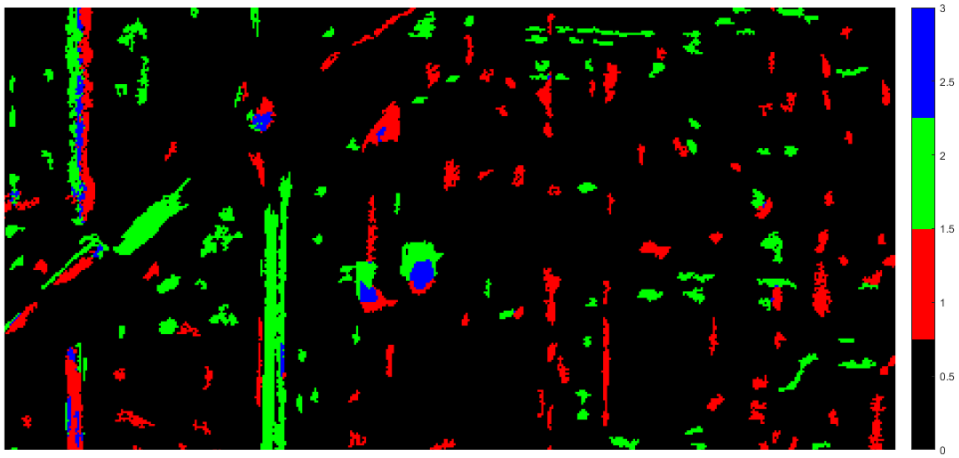


Figure 42: Display of anomalies determined from individual data sets. Red are anomalies only seen in 2015 data, green only in 2016, and blue are anomalies seen in both sets of data.

Summary

An inexpensive hyperspectral system that can be flown on small aircraft at variable intervals allows for collection of mesoscale radiometrically referenced data even over areas with no access to place spatial/spectral reference targets. These data can be used to answer a wide range of scientific questions, but this study was centered on the idea of extending the idea of detecting vegetation stress as a means of detecting a CO_2 leak from a sequestration site. Even though no CO_2 leak was present, controlled or otherwise, it was possible to use the hyperspectral data to detect local anomalies within a single data set as well as to compare multiple data sets to remove persistent anomalies and detect new anomalies. While the cause of these anomalies was not specifically investigated, some can be explained with the rudimentary knowledge of the area and show the potential of the technique.

CONCLUSION

This thesis had four major thrusts that are of direct benefit to the creation and analysis of hyperspectral data. First, a semi-automated processing workflow was developed to combine multiple swaths of hyperspectral data into a single high-resolution, mesoscale, radiometrically referenced data set without the need for ancillary instrumentation or site access. Second, biophysically relevant basis functions were developed that allowed the hyperspectral data to be analyzed based on spectral features instead of spectral bands. Third, a novel histogram based unsupervised classification technique was developed that allows a natural splitting of the data, based on the biophysically relevant parameters. Fourth, initial analysis was done on the multi-temporal data (three data sets over three years) for anomaly detection and qualitative verification of data quality. These thrusts lay the foundation for commercial software development and allow for research to address new problems, while using hyperspectral data more naturally and efficiently.

A semi-automated processing workflow was developed to combine multiple swaths of hyperspectral data into a single high-resolution, mesoscale, radiometrically referenced data set without the need for ancillary instrumentation or site access. This technique allowed for multiple flight swaths to be mosaicked into a radiometrically self-consistent data set by correcting for illumination variation between swaths. Radiometric referencing was done via selective spatial and spectral resampling of the hyperspectral data to match Landsat 8 surface reflectance data (LaSRC data product) by linearly scaling the hyperspectral data. This bootstrapping method leverages the power of the atmospheric

model used for the LaSRC data allowing a non-technical user to generate the data. Ground based measurements with a hand held hyperspectral sensor were used to verify the accuracy of the technique.

To handle hyperspectral data more naturally, a set of biophysically relevant basis functions was developed. These basis functions were designed with vegetation in mind and focused on the red edge transition and the green peak. Initial work was performed, using AVIRIS data, to show that this idea can work in urban environments as well, though further development is needed to fully utilize the data. The idea of using basis functions to represent the data as opposed to discrete spectral channels is based on Fourier transforms and quantum mechanics as they are examples where the choice of a specific basis makes analysis significantly simpler. The use of these biophysically relevant basis functions also reduces the noise and the total data size.

The type of analysis performed depends highly on the specific question of interest. One question addressed in this work had to do with classification of data that had no reference data associated with it. This lack of reference data necessitated an unsupervised classification. Most unsupervised classification routines currently (and commercially) available were developed to handle multi-spectral data causing a subtle bias in that each of the bands is given equal weight, this may not be appropriate for hyperspectral data as noise will vary with wavelength. Additionally, the classification routines based on Euclidian distance have a natural tendency to try and force clusters to contain nearly the same number of members. Using AVIRIS data with associated ground truth data it was shown that using current unsupervised classification routines improved results were obtained when using the

biophysically relevant parameters as opposed to the individual spectral bands. Additionally, the histogram based unsupervised classification routine described in this thesis did better, even though the sample size was small.

The final portion of this thesis was focused on anomaly detection as a proxy for CO_2 leak detection. Anomalies were defined based on differences in biophysical parameters from the surrounding local area and could be tracked over multiple data sets. While reference data was lacking due to site access specific examples were used to illustrate the technique. Examples were determined based on examination of the hyperspectral data as well as outside resources that inform about the land use. These examples gave qualitative support of the method, but further development would be required to develop the method. Additionally, a denser time series would allow for the detection of temporal anomalies (as opposed to spatial anomalies) and would be a more accurate representation of the type of monitoring proposed. Unfortunately, this was not an option for the area of interest under the grant for this work.

This work extends the hyperspectral field by leveraging satellites and their high-quality data and post-processing to radiometrically reference hyperspectral data. This referenced data can be used to quantitatively compare distinct data sets across space and time opening up new paths for future research. Additionally, the idea of using basis functions to represent the biophysics of the plants offers a new way of working with the ‘big data’ associated with hyperspectral imaging. The development of new tools and techniques will be required to access the full scope of information present in hyperspectral data, but the work in this thesis begins to address some of these needs.

Future work could focus on development of different basis functions for working with hyperspectral data, especially if the area of interest is urban or mining/soil based. Further work is need on the histogram splitting method to determine more accurately the natural splittings of parameters. This should center on refining the splitting rules and better determining bounds on the fit parameters to remove false peaks at the extreme edges of the histograms. Other avenues of research could focus on anomalies in agriculture fields; specifically, dealing with plant stress due to insufficient water, nitrogen, or possibly early disease detection. Eventually, much of this work could be integrated into commercial software platforms.

REFERENCES CITED

Abdalla, M. M., & El-Khoshiban, N. H. (2007). The influence of water stress on growth, relative water content, photosynthetic pigments, some metabolic and hormonal contents of two Triticium aestivum cultivars. *Journal of Applied Sciences Research*, *3*(12), 2062-2074.

Anjum, S. A., Xie, X. Y., Wang, L. C., Saleem, M. F., Man, C., & Lei, W. (2011). Morphological, physiological and biochemical responses of plants to drought stress. *African Journal of Agricultural Research*, *6*(9), 2026-2032.

Apan, A., Held, A., Phinn, S., & Markley, J. (2004). Detecting sugarcane 'orange rust' disease using EO-1 Hyperion hyperspectral imagery. *International journal of remote sensing*, *25*(2), 489-498.

Ashraf, M., & Harris, P. J. C. (2013). Photosynthesis under stressful environments: an overview. *Photosynthetica*, *51*(2), 163-190.

AVIRIS Airborne Visible/InfraRed Imaging Spectrometer. Retrieved February 07, 2017, from http://aviris.jpl.nasa.gov/aviris/

Barsi, J. A., Lee, K., Kvaran, G., Markham, B. L., & Pedelty, J. A. (2014). The spectral response of the Landsat-8 operational land imager. *Remote Sensing*, *6*(10), 10232-10251.

Bateson, L., Vellico, M., Beaubien, S. E., Pearce, J. M., Annunziatellis, A., Ciotoli, G., ... & Marsh, S. (2008). The application of remote-sensing techniques to monitor CO 2-storage sites for surface leakage: method development and testing at Latera (Italy) where naturally produced CO 2 is leaking to the atmosphere. *International Journal of Greenhouse Gas Control*, *2*(3), 388-400.

Baumgardner, M., Biehl, L., & Landgrebe, D. (2015). 220 band aviris hyperspectral image data set: June 12, 1992 indian pine test site 3. *Purdue University Research Repository*.

Behmann, J., Steinrücken, J., & Plümer, L. (2014). Detection of early plant stress responses in hyperspectral images. *ISPRS Journal of Photogrammetry and Remote Sensing*, *93*, 98-111.

Bellante, G. J., Powell, S. L., Lawrence, R. L., Repasky, K. S., & Dougher, T. A. O. (2013). Aerial detection of a simulated CO 2 leak from a geologic sequestration site using hyperspectral imagery. *International Journal of Greenhouse Gas Control*, *13*, 124-137.

Bergfeld, D., Evans, W. C., Howle, J. F., & Farrar, C. D. (2006). Carbon dioxide emissions from vegetation-kill zones around the resurgent dome of Long Valley caldera, eastern California, USA. *Journal of volcanology and Geothermal Research*, *152*(1), 140-156.

Berk, A., Anderson, G. P., Acharya, P. K., Bernstein, L. S., Muratov, L., Lee, J., ... & Lockwood, R. B. (2005, June). MODTRAN 5: a reformulated atmospheric band model with auxiliary species and practical multiple scattering options: update. In *Defense and Security* (pp. 662-667). International Society for Optics and Photonics.

Big Sky Carbon Sequestration Partnership. Retrieved February 07, 2017, from http://www.bigskyco2.org/

Blackburn, G. A. (2007). Hyperspectral remote sensing of plant pigments. *Journal of experimental botany*, *58*(4), 855-867.

Boardman, J. W. (1993). Automating spectral unmixing of AVIRIS data using convex geometry concepts.

Boardman, J. W., Kruse, F. A., & Green, R. O. (1995). Mapping target signatures via partial unmixing of AVIRIS data.

Bruegge, C. J., Stiegman, A. E., Rainen, R. A., & Springsteen, A. W. (1993). Use of Spectralon as a diffuse reflectance standard for in-flight calibration of earth-orbiting sensors. *Optical Engineering*, *32*(4), 805-814.

Byrne, G. F., Crapper, P. F., & Mayo, K. K. (1980). Monitoring land-cover change by principal component analysis of multitemporal Landsat data. *Remote Sensing of Environment*, *10*(3), 175-184.

Carlson, T. N., & Ripley, D. A. (1997). On the relation between NDVI, fractional vegetation cover, and leaf area index. *Remote sensing of Environment*, *62*(3), 241-252.

Carter, G. A. (1991). Primary and secondary effects of water content on the spectral reflectance of leaves. *American Journal of Botany*, 916-924.

Carter, G. A., Mitchell, R. J., Chappelka, A. H., & Brewer, C. H. (1992). Response of leaf spectral reflectance in loblolly pine to increased atmospheric ozone and precipitation acidity. *Journal of experimental botany*, *43*(4), 577-584.

Carter, G. A., & Knapp, A. K. (2001). Leaf optical properties in higher plants: linking spectral characteristics to stress and chlorophyll concentration. *American Journal of Botany*, *88*(4), 677-684.

Carter, G. A., & Young, D. R. (1993). Foliar spectral reflectance and plant stress on a barrier island. *International Journal of Plant Sciences*, 298-305.

Cha, Y., & Kim, S. (2007). The error-amended sharp edge (EASE) scheme for image zooming. *IEEE Transactions on Image Processing*, *16*(6), 1496-1505.

Chapin, F. S. (1991). Integrated responses of plants to stress. *BioScience*, *41*(1), 29-36.

Chen, P., Nelson, J. D., & Tourneret, J. Y. (2017). Toward a Sparse Bayesian Markov Random Field Approach to Hyperspectral Unmixing and Classification. *IEEE Transactions on Image Processing*, *26*(1), 426-438.

Cho, M. A., & Skidmore, A. K. (2006). A new technique for extracting the red edge position from hyperspectral data: The linear extrapolation method. *Remote sensing of environment*, *101*(2), 181-193.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and psychological measurement*, *20*(1), 37-46.

Cooley, T., Anderson, G. P., Felde, G. W., Hoke, M. L., Ratkowski, A. J., Chetwynd, J. H., ... & Bernstein, L. S. (2002). FLAASH, a MODTRAN4-based atmospheric correction algorithm, its application and validation. In *Geoscience and Remote Sensing Symposium, 2002. IGARSS'02. 2002 IEEE International* (Vol. 3, pp. 1414-1418). IEEE.

Dawson, T. P., & Curran, P. J. (1998). Technical note A new technique for interpolating the reflectance red edge position.

DeFries, R. S., & Townshend, J. R. G. (1994). NDVI-derived land cover classifications at a global scale. *International Journal of Remote Sensing*, *15*(17), 3567-3586.

Demirel, N., Emil, M. K., & Duzgun, H. S. (2011). Surface coal mine area monitoring using multi-temporal high-resolution satellite imagery. *International journal of Coal geology*, *86*(1), 3-11.

Dobrowski, S. Z., Pushnik, J. C., Zarco-Tejada, P. J., & Ustin, S. L. (2005). Simple reflectance indices track heat and water stress-induced changes in steady-state chlorophyll fluorescence at the canopy scale. *Remote Sensing of Environment*, *97*(3), 403-414.

Eismann, M. T. (2012, April). Hyperspectral remote sensing. Bellingham: SPIE.

Eitel, J. U., Vierling, L. A., Litvak, M. E., Long, D. S., Schulthess, U., Ager, A. A., ... & Stoscheck, L. (2011). Broadband, red-edge information from satellites improves early stress detection in a New Mexico conifer woodland. *Remote Sensing of Environment*, *115*(12), 3640-3646.

Elachi, C., & Van Zyl, J. J. (2006). *Introduction to the physics and techniques of remote sensing* (Vol. 28). John Wiley & Sons.

Fattal, R. (2007, August). Image upsampling via imposed edge statistics. In *ACM Transactions on Graphics (TOG)* (Vol. 26, No. 3, p. 95). ACM.

Folkman, M. A., Pearlman, J., Liao, L. B., & Jarecke, P. J. (2001, February). EO-1/Hyperion hyperspectral imager design, development, characterization, and calibration. In *Second International Asia-Pacific Symposium on Remote Sensing of the Atmosphere, Environment, and Space* (pp. 40-51). International Society for Optics and Photonics.

Franke, J., & Menz, G. (2007). Multi-temporal wheat disease detection by multi-spectral remote sensing. *Precision Agriculture*, *8*(3), 161-172.

Galvao, L. S., Formaggio, A. R., & Tisot, D. A. (2005). Discrimination of sugarcane varieties in Southeastern Brazil with EO-1 Hyperion data. *Remote Sensing of Environment*, *94*(4), 523-534.

Gao, J. (2008). *Digital analysis of remotely sensed imagery*. McGraw-Hill Professional.

Gates, D. M., Keegan, H. J., Schleter, J. C., & Weidner, V. R. (1965). Spectral properties of plants. *Applied optics*, *4*(1), 11-20.

Gevaert, C. M., Suomalainen, J., Tang, J., & Kooistra, L. (2015). Generation of Spectral–Temporal Response Surfaces by Combining Multispectral Satellite and Hyperspectral UAV Imagery for Precision Agriculture Applications. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *8*(6), 3140-3146.

Gitelson, A. A., Merzlyak, M. N., & Lichtenthaler, H. K. (1996). Detection of red edge position and chlorophyll content by reflectance measurements near 700 nm. *Journal of Plant Physiology*, *148*(3-4), 501-508.

Goenaga, M. A., Torres-Madronero, M. C., Velez-Reyes, M., Van Bloem, S. J., & Chinea, J. D. (2013). Unmixing analysis of a time series of Hyperion images over the Guánica dry forest in Puerto Rico. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *6*(2), 329-338.

Gomez-Chova, L., Fernández-Prieto, D., Calpe, J., Soria, E., Vila, J., & Camps-Valls, G. (2006). Urban monitoring using multi-temporal SAR and multi-spectral data. *Pattern Recognition Letters*, *27*(4), 234-243.

Govender, M., Chetty, K., & Bulcock, H. (2007). A review of hyperspectral remote sensing and its application in vegetation and water resource studies. *Water Sa*, *33*(2), 145-151.

Green, R. (2001). Atmospheric correction now (ACORN). *developed by ImSpec LLC, available from Analytical Imaging and Geophysics LLC*.

Gueymard, C. A. (2005). SMARTS Code, Version 2.9. 5 User's Manual. *Solar Consulting Services. Available from http://www.solarconsultingservices.com/SMARTS295_manual.pdf.*

Hadjimitsis, D. G., Clayton, C. R. I., & Retalis, A. (2009). The use of selected pseudo-invariant targets for the application of atmospheric correction in multi-temporal studies using satellite remotely sensed imagery. *International Journal of Applied Earth Observation and Geoinformation*, *11*(3), 192-200.

Hall, A., Lamb, D. W., Holzapfel, B., & Louis, J. (2002). Optical remote sensing applications in viticulture- a review. *Australian Journal of Grape and Wine Research*, *8*(1), 36-47.

Hemissi, S., Farah, I. R., Ettabaa, K. S., & Solaiman, B. (2013). Multi-spectro-temporal analysis of hyperspectral imagery based on 3-D spectral modeling and multilinear algebra. *IEEE Transactions on Geoscience and Remote Sensing*, *51*(1), 199-216.

Horler, D. N. H., Dockray, M., & Barber, J. (1983). The red edge of plant leaf reflectance. *International Journal of Remote Sensing*, *4*(2), 273-288.

Howell, D. C. (2014). Median absolute deviation. *Wiley StatsRef: Statistics Reference Online*.

Huete, A. R., Miura, T., & Gao, X. (2003). Land cover conversion and degradation analyses through coupled soil-plant biophysical parameters derived from hyperspectral EO-1 Hyperion. *IEEE Transactions on Geoscience and Remote Sensing*, *41*(6), 1268-1276.

HyspIRI Mission Study. Retrieved February 07, 2017, from https://hyspiri.jpl.nasa.gov/

Jackson, R. D., Clarke, T. R., & Moran, M. S. (1992). Bidirectional calibration results for 11 Spectralon and 16 BaSO4 reference reflectance panels. *Remote Sensing of Environment*, *40*(3), 231-239.

Jaleel, C. A., Manivannan, P. A., Wahid, A., Farooq, M., Al-Juburi, H. J., Somasundaram, R., & Panneerselvam, R. (2009). Drought stress in plants: a review on morphological characteristics and pigments composition. *Int J Agric Biol*, *11*(1), 100-105.

Jones, C. L., Weckler, P. R., Maness, N. O., Stone, M. L., & Jayasekara, R. (2004). Estimating water stress in plants using hyperspectral sensing. In *2004 ASAE Annual Meeting* (p. 1). American Society of Agricultural and Biological Engineers.

Keith, C. J., Repasky, K. S., Lawrence, R. L., Jay, S. C., & Carlsten, J. L. (2009). Monitoring effects of a controlled subsurface carbon dioxide release on vegetation using a hyperspectral imager. *International Journal of Greenhouse Gas Control*, *3*(5), 626-632.

Kerekes, J., Muldowney, M., Strackerjan, K., Smith, L., & Leahy, B. (2006, May). Vehicle tracking with multi-temporal hyperspectral imagery. In *Defense and Security Symposium* (pp. 62330C-62330C). International Society for Optics and Photonics.

Keshava, N., & Mustard, J. F. (2002). Spectral unmixing. *IEEE signal processing magazine*, *19*(1), 44-57.

Keshava, N. (2003). A survey of spectral unmixing algorithms. *Lincoln Laboratory Journal*, *14*(1), 55-78.

Kim, Y., Glenn, D. M., Park, J., Ngugi, H. K., & Lehman, B. L. (2011). Hyperspectral image analysis for water stress detection of apple trees. *Computers and Electronics in Agriculture*, *77*(2), 155-160.

Knight, E. J., & Kvaran, G. (2014). Landsat-8 operational land imager design, characterization and performance. *Remote Sensing*, *6*(11), 10286-10305.

Knipling, E. B. (1970). Physical and physiological basis for the reflectance of visible and near-infrared radiation from vegetation. *Remote sensing of environment*, *1*(3), 155-159.

Korbøl, R., & Kaddour, A. (1995). Sleipner vest CO 2 disposal-injection of removed CO 2 into the Utsira formation. *Energy Conversion and Management*, *36*(6), 509-512.

Kruse, F. A., Boardman, J. W., & Huntington, J. F. (2003). Comparison of airborne hyperspectral data and EO-1 Hyperion for mineral mapping. *IEEE Transactions on Geoscience and Remote Sensing*, *41*(6), 1388-1400.

Kuon, I., Tessier, R., & Rose, J. (2008). FPGA architecture: Survey and challenges. *Foundations and Trends in Electronic Design Automation*, *2*(2), 135-253.

Lausch, A., Pause, M., Schmidt, A., Salbach, C., Gwillym-Margianto, S., & Merbach, I. (2013). Temporal hyperspectral monitoring of chlorophyll, LAI, and water content of barley during a growing season. *Canadian Journal of Remote Sensing*, *39*(3), 191-207.

Lawrence, R., & Labus, M. (2003). Early detection of Douglas-fir beetle infestation with subcanopy resolution hyperspectral imagery. *Western Journal of Applied Forestry*, *18*(3), 202-206.

Lee, Y. J., & Yoon, J. (2010). Nonlinear image upsampling method based on radial basis function interpolation. *IEEE Transactions on Image Processing*, *19*(10), 2682-2692.

Leys, C., Ley, C., Klein, O., Bernard, P., & Licata, L. (2013). Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median. *Journal of Experimental Social Psychology*, *49*(4), 764-766.

Li, L., Ustin, S. L., & Lay, M. (2005). Application of AVIRIS data in detection of oil-induced vegetation stress and cover change at Jornada, New Mexico. *Remote Sensing of Environment*, *94*(1), 1-16.

Li, X., & Orchard, M. T. (2001). New edge-directed interpolation. *IEEE transactions on image processing*, *10*(10), 1521-1527.

Li, X., & Yeh, A. G. O. (1998). Principal component analysis of stacked multi-temporal images for the monitoring of rapid urban expansion in the Pearl River Delta. *International Journal of Remote Sensing*, *19*(8), 1501-1518.

Liangyun, L., Muyi, H., & Wenjiang, H. (2004). Monitoring stripe rust disease of winter wheat using multi-temporal hyperspectral airborne data. *Journal of Remote Sensing*, *8*(3), 275-281.

Lichtenthaler, H. K. (1987). [34] Chlorophylls and carotenoids: pigments of photosynthetic biomembranes. *Methods in enzymology*, *148*, 350-382.

Liu, D., Kelly, M., & Gong, P. (2006). A spatial–temporal approach to monitoring forest disease spread using multi-temporal high spatial resolution imagery. *Remote sensing of environment*, *101*(2), 167-180.

Liu, J., Pattey, E., Miller, J. R., McNairn, H., Smith, A., & Hu, B. (2010). Estimating crop stresses, aboveground dry biomass and yield of corn using multi-temporal optical data combined with a radiation use efficiency model. *Remote Sensing of Environment*, *114*(6), 1167-1177.

Lunetta, R. S., Knight, J. F., Ediriwickrema, J., Lyon, J. G., & Worthy, L. D. (2006). Land-cover change detection using multi-temporal MODIS NDVI data. *Remote sensing of environment*, *105*(2), 142-154.

Luther, J. E., & Carroll, A. L. (1999). Development of an index of balsam fir vigor by foliar spectral reflectance. *Remote sensing of Environment*, *69*(3), 241-252.

Maček, I., Pfanz, H., Francetič, V., Batič, F., & Vodnik, D. (2005). Root respiration response to high CO 2 concentrations in plants from natural CO 2 springs. *Environmental and Experimental Botany*, *54*(1), 90-99.

Maldal, T., & Tappel, I. M. (2004). CO 2 underground storage for Snøhvit gas field development. *Energy*, *29*(9), 1403-1411.

Male, E. J., Pickles, W. L., Silver, E. A., Hoffmann, G. D., Lewicki, J., Apple, M., ... & Burton, E. A. (2010). Using hyperspectral plant signatures for CO2 leak detection during the 2008 ZERT CO2 sequestration field experiment in Bozeman, Montana. *Environmental Earth Sciences*, *60*(2), 251-261.

Markham, B., Barsi, J., Kvaran, G., Ong, L., Kaita, E., Biggar, S., ... & Helder, D. (2014). Landsat-8 operational land imager radiometric calibration and stability. *Remote Sensing*, *6*(12), 12275-12308.

Maktav, D., & Erbek, F. S. (2005). Analysis of urban growth using multi- temporal satellite data in Istanbul, Turkey. *International Journal of Remote Sensing*, *26*(4), 797-810.

Miller, J. R., Hare, E. W., & Wu, J. (1990). Quantitative characterization of the vegetation red edge reflectance 1. An inverted-Gaussian reflectance model. *Remote Sensing*, *11*(10), 1755-1773.

Nguyen, H. T., & Lee, B. W. (2006). Assessment of rice leaf growth and nitrogen status by hyperspectral canopy reflectance and partial least square regression. *European Journal of Agronomy*, *24*(4), 349-356.

Noomen, M. F., & Skidmore, A. K. (2009). The effects of high soil CO2 concentrations on leaf reflectance of maize plants. *International Journal of Remote Sensing*, *30*(2), 481-497.

Noomen, M. F., Smith, K. L., Colls, J. J., Steven, M. D., Skidmore, A. K., & Van Der Meer, F. D. (2008). Hyperspectral indices for detecting changes in canopy reflectance as a result of underground natural gas leakage. *International Journal of Remote Sensing*, *29*(20), 5987-6008.

Nowatzki, J. F., Andres, R., & Kyllo, K. (2004). *Agricultural remote sensing basics*. NDSU Extension Service.

Ong, C., & Cudahy, T. (2002, March). Deriving quantitative monitoring data related to acid drainage using multi-temporal hyperspectral data. In *AVIRIS Workshop*.

Pearlman, J., Carman, S., Segal, C., Jarecke, P., Clancy, P., & Browne, W. (2001). Overview of the Hyperion imaging spectrometer for the NASA EO-1 mission. In *Geoscience and Remote Sensing Symposium, 2001. IGARSS'01. IEEE 2001 International* (Vol. 7, pp. 3036-3038). IEEE.

Pengra, B. W., Johnston, C. A., & Loveland, T. R. (2007). Mapping an invasive plant, Phragmites australis, in coastal wetlands using the EO-1 Hyperion hyperspectral sensor. *Remote Sensing of Environment*, *108*(1), 74-81.

Peñuelas, J., Filella, I., Biel, C., Serrano, L., & Save, R. (1993). The reflectance at the 950–970 nm region as an indicator of plant water status. *International journal of remote sensing*, *14*(10), 1887-1905.

Petitjean, F., Kurtz, C., Passat, N., & Gançarski, P. (2012). Spatio-temporal reasoning for the classification of satellite image time series. *Pattern Recognition Letters*, *33*(13), 1805-1815.

Pickles, W., & Cover, W. (2004). *Hyperspectral geobotanical remote sensing for CO2 storage monitoring* (No. UCRL-BOOK-204165). Lawrence Livermore National Laboratory (LLNL), Livermore, CA.

Purkis, S., & Klemas, V. Remote sensing basics. *Remote Sensing and Global Environmental Change*, 6-17.

Rao, C. R. (2009). *Linear statistical inference and its applications* (Vol. 22). John Wiley & Sons.

Regional partnerships. Retrieved February 7, 2017, from https://energy.gov/fe/science-innovation/carbon-capture-and-storage-research/regional-partnerships

Richter, R., & Schlapfer, D. (2012). Atmospheric/topographic correction for airborne imagery: ATCOR-4 User Guide. *DLR IB*, 565-02.

Roberts, D., Alonzo, M., Wetherley, E. B., Dudley, K. L., & Dennison, P. E. (2017). Multiscale Analysis of Urban Areas Using Mixing Models. *Integrating Scale in Remote Sensing and GIS*, 247.

Rogan, J., Franklin, J., & Roberts, D. A. (2002). A comparison of methods for monitoring multitemporal vegetation change using Thematic Mapper imagery. *Remote Sensing of Environment*, *80*(1), 143-156.

Rowan, K. S. (1989). *Photosynthetic pigments of algae*. CUP Archive.

Roy, D. P., Wulder, M. A., Loveland, T. R., Woodcock, C. E., Allen, R. G., Anderson, M. C., ... & Scambos, T. A. (2014). Landsat-8: Science and product vision for terrestrial global change research. *Remote Sensing of Environment*, *145*, 154-172.

Rutqvist, J., Vasco, D. W., & Myer, L. (2010). Coupled reservoir-geomechanical analysis of CO 2 injection and ground deformations at In Salah, Algeria. *International Journal of Greenhouse Gas Control*, *4*(2), 225-230.

Santosh, K. M., & Sundaresan, J. (2014). Remote Sensing Basics. In *Geospatial Technologies and Climate Change* (pp. 279-290). Springer International Publishing.

Senthilnath, J., Omkar, S. N., Mani, V., Karnwal, N., & Shreyas, P. B. (2013). Crop stage classification of hyperspectral data using unsupervised techniques. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *6*(2), 861-866.

Shippert, P. (2003). Introduction to hyperspectral image analysis. *Online Journal of Space Communication*, *3*.

Sims, D. A., & Gamon, J. A. (2002). Relationships between leaf pigment content and spectral reflectance across a wide range of species, leaf structures and developmental stages. *Remote sensing of environment*, *81*(2), 337-354.

Smith, K. L., Steven, M. D., & Colls, J. J. (2004). Use of hyperspectral derivative ratios in the red-edge region to identify plant stress responses to gas leaks. *Remote sensing of environment*, *92*(2), 207-217.

Smith, W. K., Vogelmann, T. C., DeLucia, E. H., Bell, D. T., & Shepherd, K. A. (1997). Leaf form and photosynthesis. *Bioscience*, *47*(11), 785-793.

Somers, B., & Asner, G. P. (2012). Hyperspectral time series analysis of native and invasive species in Hawaiian rainforests. *Remote Sensing*, *4*(9), 2510-2529.

Somers, B., & Asner, G. P. (2013). Multi-temporal hyperspectral mixture analysis and feature selection for invasive species mapping in rainforests. *Remote Sensing of Environment*, *136*, 14-27.

Standard, A. S. T. M. (2007). G173,"Standard Tables for Reference Solar Spectral Irradiances: Direct Normal and Hemispherical on 37 Tilted Surface," Amer. *Society for Testing Matls., West Conshocken PA, USA*.

Storey, J., Choate, M., & Lee, K. (2014). Landsat 8 Operational Land Imager on-orbit geometric calibration and performance. *Remote sensing*, *6*(11), 11127-11152.

Strachan, I. B., Pattey, E., & Boisvert, J. B. (2002). Impact of nitrogen and environmental conditions on corn as detected by hyperspectral reflectance. *Remote Sensing of Environment*, *80*(2), 213-224.

Stuckens, J., Dzikiti, S., Verstraeten, W. W., Verreynne, S., Swennen, R., & Coppin, P. (2011). Physiological interpretation of a hyperspectral time series in a citrus orchard. *Agricultural and forest meteorology*, *151*(7), 1002-1015.

Suárez, L., Zarco-Tejada, P. J., Sepulcre-Cantó, G., Pérez-Priego, O., Miller, J. R., Jiménez-Muñoz, J. C., & Sobrino, J. (2008). Assessing canopy PRI for water stress detection with diurnal airborne imagery. *Remote Sensing of Environment*, *112*(2), 560-575.

Tambussi, E. A., Casadesus, J., Munné-Bosch, S., & Araus, J. L. (2002). Photoprotection in water-stressed plants of durum wheat (Triticum turgidum var. durum): changes in chlorophyll fluorescence, spectral signature and photosynthetic pigments. *Functional Plant Biology*, *29*(1), 35-44.

Thenkabail, A., Lyon, P. S., & Huete, J. G. (2011). *Hyperspectral remote sensing of vegetation*. CRC Press.

Tilling, A. K., O'Leary, G. J., Ferwerda, J. G., Jones, S. D., Fitzgerald, G. J., Rodriguez, D., & Belford, R. (2007). Remote sensing of nitrogen and water stress in wheat. *Field Crops Research*, *104*(1), 77-85.

Travelletti, J., Delacourt, C., Allemand, P., Malet, J. P., Schmittbuhl, J., Toussaint, R., & Bastard, M. (2012). Correlation of multi-temporal ground-based optical images for landslide monitoring: Application, potential and limitations. *ISPRS Journal of Photogrammetry and Remote Sensing*, *70*, 39-55.

Tripathy, G. K., Ghosh, T. K., & Shah, S. D. (1996). Monitoring of desertification process in Karnataka state of India using multi-temporal remote sensing and ancillary information using GIS. *International Journal of Remote Sensing*, *17*(12), 2243-2257.

Vagni, F. (2007). Survey of hyperspectral and multispectral imaging technologies.

Vaughan, M. A., Liu, Z., McGill, M. J., Hu, Y., & Obland, M. D. (2010). On the spectral dependence of backscatter from cirrus clouds: Assessing CALIOP's 1064 nm calibration assumptions using cloud physics lidar measurements. *Journal of Geophysical Research: Atmospheres*, *115*(D14).

Weibel, R., & Hansman, R. J. (2005). An integrated approach to evaluating risk mitigation measures for UAV operational concepts in the NAS. In *Infotech@ Aerospace* (p. 6957).

Wells Search. Retrieved February 20, 2017, from http://www.bogc.dnrc.mt.gov/webapps/dataminer/Wells/Wells.aspx.

Whittaker, S. G. (2004). Geological storage of greenhouse gases: the IEA Weyburn CO2 monitoring and storage project. *Canadian Society of Petroleum and Geologists Reservoir*, *31*(8), 9.

Wójtowicz, M., Wójtowicz, A., & Piekarczyk, J. (2016). Application of remote sensing methods in agriculture. *Communications in Biometry and Crop Science*, *11*, 31-50.

Wright, R., Mourits, F., Rodríguez, L. B., & Serrano, M. D. (2013). The First North American Carbon Storage Atlas. *Energy Procedia*, *37*, 5280-5289.

Xian, G., Crane, M., & McMahon, C. (2008). Quantifying multi-temporal urban development characteristics in Las Vegas from Landsat and ASTER data. *Photogrammetric Engineering & Remote Sensing*, *74*(4), 473-481.

Xie, C. Q., Li, X. L., Nie, P. C., & He, Y. (2013). Application of time series hyperspectral imaging (TS-HSI) for determining water content within tea leaves during drying. *T ASABE*, *56*, 1431-1440.

Xu, X., & Shi, Z. (2017). Multi-objective based spectral unmixing for hyperspectral images. *ISPRS Journal of Photogrammetry and Remote Sensing*, *124*, 54-69.

Yuan, Y., Lv, H., & Lu, X. (2015). Semi-supervised change detection method for multi-temporal hyperspectral images. *Neurocomputing*, *148*, 363-375.

Zarco-Tejada, P. J., Miller, J. R., Mohammed, G. H., & Noland, T. L. (2000). Chlorophyll fluorescence effects on vegetation apparent reflectance: I. Leaf-level measurements and model simulation. *Remote Sensing of Environment*, *74*(3), 582-595.

Zarco-Tejada, P. J., Ustin, S. L., & Whiting, M. L. (2005). Temporal and spatial relationships between within-field yield variability in cotton and high-spatial hyperspectral remote sensing imagery. *Agronomy Journal*, *97*(3), 641-653.

Zhang, M., Qin, Z., Liu, X., & Ustin, S. L. (2003). Detection of stress in tomatoes induced by late blight disease in California, USA, using hyperspectral remote sensing. *International Journal of Applied Earth Observation and Geoinformation*, *4*(4), 295-310.

Zhang, X., & Wu, X. (2008). Image interpolation by adaptive 2-D autoregressive modeling and soft-decision estimation. *IEEE Transactions on Image Processing*, *17*(6), 887-896.

Zhao, C., Liu, L., Wang, J., Huang, W., Song, X., & Li, C. (2005). Predicting grain protein content of winter wheat using remote sensing data based on nitrogen status and water stress. *International Journal of Applied Earth Observation and Geoinformation*, *7*(1), 1-9.

Zhao, H., Jia, G., & Li, N. (2010). Transformation from hyperspectral radiance data to data of other sensors based on spectral superresolution. *IEEE Transactions on Geoscience and Remote Sensing*, *48*(11), 3903-3912.

APPENDICES

APPENDIX A

MATLAB CODE FOR SPATIAL AND SPECTRAL PROCESSING

Program with takes separate swaths as inputs saved in the working directory as 1_Optimized.tif, 2.tif, 3.tif, etc. and loops through each pair (1_Optimized & 2, 2_Optimzed & 3, etc.) and corrects the second swath to match the first. There are three degrees of freedom, Track with allows for a smoothly varying correction, Cross Track which is a linear correction perpendicular to the direction of travel, and Brightness with is a linear correction across each of the 80 bands. After the second swath is corrected it is saved to N_Optimzied.tif in the working directory. There are options for plotting various aspects during the optimization run, but for the sake of memory use these are commented out.

Most of the parameters are fixed and would not need to be changed with the exceptions of the smoothing window (currently 201 pixels), and number of bands (currently 80). These parameters could be derived within the program, but this was not done for simplicity.

```
%% Swath Matching - takes into account variation along Track, and assumes linear Cross Track model as well as correcting brightness

MaxImage = 4; %number of swaths to match
All = tic;
for n=1:(MaxImage-1) %number of overlapping areas
[image1,R1] = geotiffread([num2str(n),'_Optimized']); %input data must be labeled as 1_Optimized.tif, 2.tif, etc.
[image2,R2] = geotiffread(num2str(n+1));

%the assumption is that the lower image is the Master image
Upper = double(image2); %upper (northmost) image
Lower = double(image1);
clear image1
clear image2

%Determine limits automatically
XDiff = R1.XWorldLimits-R2.XWorldLimits;
```

```
YDiff = R1.YWorldLimits-R2.YWorldLimits;
if XDiff(1) < 0 %determines which image extends furthest to the west (left)
   ULUpperX = 1;
   ULLowerX = abs(XDiff(1))+1;
else
   ULUpperX = abs(XDiff(1))+1;
   ULLowerX = 1;
end
if YDiff(2) < 0 %determines which image extends furthest to the north (up)
   ULUpperY = abs(YDiff(2))+1;
   ULLowerY = 1;
else
   ULUpperY = 1;
   ULLowerY = abs(YDiff(2))+1;
end

%determines 'bounding box' for overlap
WidthUpper = size(Upper,2);
WidthLower = size(Lower,2);
TotalWidth = min((WidthUpper-ULUpperX), (WidthLower-ULLowerX));
HeightUpper = size(Upper,1);
HeightLower = size(Lower,1);
TotalHeight = min((HeightUpper-ULUpperY), (HeightLower-ULLowerY));

FullUpperSubset = Upper((ULUpperY):(ULUpperY+TotalHeight),
(ULUpperX):(ULUpperX+TotalWidth), :);
FullUpperSubset(FullUpperSubset < 1.1) = NaN; %null out 0's
FullLowerSubset = Lower((ULLowerY):(ULLowerY+TotalHeight),
(ULLowerX):(ULLowerX+TotalWidth), :);
FullLowerSubset(FullLowerSubset < 1.1) = NaN; %null out 0's

disp([num2str(sum(~isnan(FullLowerSubset(:)))/size(FullLowerSubset,3)), ' total overlap
points.'])
options = optimoptions(@fminunc, 'MaxFunEvals', 10000, 'Display', 'off', 'TolFun',1e-6);

tic

resizefactor = 1/3; %reduces image size in both axis by this factor
UpperBlur = imresize(FullUpperSubset, resizefactor);
LowerBlur = imresize(FullLowerSubset, resizefactor);
disp([num2str(sum(~isnan(LowerBlur(:)))/size(LowerBlur,3)), ' points being evaluated,
be patient.'])

SaveFit = zeros(size(UpperBlur,1), size(UpperBlur,2), 2);
```

```
parfor_progress(size(UpperBlur,1));
for Y = 1:size(UpperBlur,1)
    temp = zeros(1,size(UpperBlur,2),2);
    parfor X = 1:size(UpperBlur,2)
        coeffs = polyfit(squeeze(UpperBlur(Y,X,:)), squeeze(LowerBlur(Y,X,:)), 1);
        temp(1,X,:) = coeffs;
    end
    parfor_progress;
    SaveFit(Y,:,:) = temp(1,:,:);
end
parfor_progress(0);

if 0 %turn on to plot slope and intercept of overlap areas
    figure(1)
    clf(1)
    subplot(2,1,1)
    surface(SaveFit(:,:,1),'EdgeColor','none')
    caxis([prctile(reshape(SaveFit(:,:,1), 1, numel(SaveFit(:,:,1))), 1),
prctile(reshape(SaveFit(:,:,1), 1, numel(SaveFit(:,:,1))), 99)])
    colorbar
    colormap(jet)
    subplot(2,1,2)
    surface(SaveFit(:,:,2),'EdgeColor','none')
    caxis([prctile(reshape(SaveFit(:,:,2), 1, numel(SaveFit(:,:,2))), 1),
prctile(reshape(SaveFit(:,:,2), 1, numel(SaveFit(:,:,2))), 99)])
    colorbar
    colormap(jet)
end

SlopeMean = nanmean(SaveFit(:,:,1)); %Deals with slope
SlopeMean(isnan(SlopeMean)) = nanmean(SlopeMean); %just in case any NaN's sneak
through
NewSlopeMean = smooth(SlopeMean,201,'rloess'); %this is a fairly large window, but it
seems to work

if 0 %turn on to plot slope and intercepts along with smoothed version
    figure(2)
    clf(2)
    subplot(2,1,1)
    hold on
    plot(SlopeMean,'c-', 'LineWidth', 2)
    plot(NewSlopeMean,'b-', 'LineWidth', 2)
    % title('Mean Track Slope', 'FontSize', 32)
    xlabel('Scaled Distance (m)', 'FontSize', 24)
```

```
    ylabel('Mean Track Gain', 'FontSize', 24)
    axis([0,size(SlopeMean,2), prctile(SlopeMean,1), prctile(SlopeMean,99)])
    set(gca,'fontsize',24)
    hold off

    InterMean = nanmean(SaveFit(:,:,2)); %Deals with Intercept
    InterMean(isnan(InterMean)) = nanmean(InterMean);
    NewInterMean = smooth(InterMean,201,'rloess');
    subplot(2,1,2)
    hold on
    plot(InterMean,'c-', 'LineWidth', 2)
    plot(NewInterMean,'b-', 'LineWidth', 2)
    % title('Mean Track Intercept', 'FontSize', 32)
    xlabel('Scaled Distance (m)', 'FontSize', 24)
    ylabel('Mean Track Bias', 'FontSize', 24)
    axis([0,size(InterMean,2), prctile(InterMean,1), prctile(InterMean,99)])
    set(gca,'fontsize',24)
    hold off
    drawnow
    % clear SaveFit
end

NewUpper = repmat(interp1((ULUpperX):(1/resizefactor):(ULUpperX+TotalWidth),
NewSlopeMean, 1:WidthUpper, 'pchip', mean(NewSlopeMean)), [HeightUpper,1,80]) .*
Upper +
repmat(interp1((ULUpperX):(1/resizefactor):(ULUpperX+TotalWidth),NewInterMean,
1:WidthUpper, 'pchip', mean(NewInterMean)), [HeightUpper,1,80]);
NewFullUpperSubset = NewUpper((ULUpperY):(ULUpperY+TotalHeight),
(ULUpperX):(ULUpperX+TotalWidth), :);
NewFullUpperSubset(isnan(FullUpperSubset)) = NaN; %null out 0's
NewUpperBlur = imresize(NewFullUpperSubset, resizefactor);

Initial = OverlapBrightnessSimple(LowerBlur, UpperBlur, [1,1], [0,1,0,0]);
TI = OverlapBrightnessSimple(LowerBlur, NewUpperBlur, [1,1], [0,1,0,0]);
disp(['Track/Intensity improved overlap by ', num2str((Initial-TI)/Initial*100), '%.'])

%Correct for Cross Track issues
[xCT] = fminsearch(@(x)OverlapBrightnessSimple(LowerBlur, NewUpperBlur, [1,0],
x), [1e-5,1,1,0], options);
CT = OverlapBrightnessSimple(LowerBlur, NewUpperBlur, [1,0], xCT);
disp(['Track/Intensity/CrossTrack improved overlap by ', num2str((Initial-
CT)/Initial*100), '%.'])

Dim = size(NewUpper);
```

```
for Band = 1:80
   NewUpper(:,:,Band) =
NewUpper(:,:,Band).*repmat(xCT(1)*resizefactor*((size(NewFullUpperSubset,1)-
Dim(1)+1):(size(NewFullUpperSubset,1)))'+xCT(2),1,Dim(2));
end
NewFullUpperSubset = NewUpper((ULUpperY):(ULUpperY+TotalHeight),
(ULUpperX):(ULUpperX+TotalWidth), :);
NewFullUpperSubset(isnan(FullUpperSubset)) = NaN;
ScaledUpperBlur = imresize(NewFullUpperSubset, resizefactor);

%Make final intensity correction, mainly since Cross Track may have changed the values
slightly.
[xI] = fminsearch(@(x)OverlapBrightnessSimple(LowerBlur, ScaledUpperBlur, [0,1], x),
[1e-5,1,1,0], options);
I = OverlapBrightnessSimple(LowerBlur, ScaledUpperBlur, [0,1], xI);
disp(['Track/Intensity/CrossTrack/Intensity improved overlap by ', num2str((Initial-
I)/Initial*100), '%.'])

for Band = 1:80
   NewUpper(:,:,Band) = xI(3)*NewUpper(:,:,Band)+xI(4);
end
toc
if 0 %turn on to plot differences before and after correction for the overlap area.  Some
options for display are commented out.
   figure(4)
   clf(4)
   subplot(2,1,1)
   surface(sum(abs(LowerBlur-UpperBlur),3)/80,'EdgeColor','none')
   caxis([0, prctile(reshape(sum(abs(LowerBlur-UpperBlur),3), 1,
numel(sum(LowerBlur-UpperBlur,3))), 90)/80])
   axis([1,size(UpperBlur,2), 1, size(UpperBlur,1)])
   xlabel('Scaled Distance (m)')
   ylabel('Scaled Distance (m)')
   set(gca,'fontsize',24)
   % title(['Before Correction.  Mean offset of ',
num2str(round(nanmean(nanmean(sum(abs(LowerBlur-UpperBlur),3)/80)))), ' per pixel
per band'], 'FontSize', 20)
   colorbar
   % subplot(3,1,2)
   % surface(sum(abs(LowerBlur-NewUpperBlur),3)/80,'EdgeColor','none')
   % title(['After Track/Intensity Correction. Mean offset of ',
num2str(nanmean(nanmean(sum(abs(LowerBlur-NewUpperBlur),3)/80))), ' per pixel per
band'])
```

```
    % caxis([0, prctile(reshape(sum(abs(LowerBlur-UpperBlur),3), 1,
numel(sum(LowerBlur-UpperBlur,3))), 95)/80])
    % axis([1,size(UpperBlur,2), 1, size(UpperBlur,1)])
    % colorbar
    subplot(2,1,2)
    % Needed for plotting Track/Intensity/CrossTrack/Intensity Correction
    NewFullUpperSubset = NewUpper((ULUpperY):(ULUpperY+TotalHeight),
(ULUpperX):(ULUpperX+TotalWidth), :);
    NewFullUpperSubset(isnan(FullUpperSubset)) = NaN;
    ScaledUpperBlur = imresize(NewFullUpperSubset, resizefactor);
    %Clear out data before plotting
    clear NewFullUpperSubset
    surface(sum(abs(LowerBlur-ScaledUpperBlur),3)/80,'EdgeColor','none')
    caxis([0, prctile(reshape(sum(abs(LowerBlur-UpperBlur),3), 1,
numel(sum(LowerBlur-UpperBlur,3))), 90)/80])
    axis([1,size(UpperBlur,2), 1, size(UpperBlur,1)])
    xlabel('Scaled Distance (m)')
    ylabel('Scaled Distance (m)')
    set(gca,'fontsize',24)
    % title(['After Track/Intensity/CrossTrack/Intensity Correction.  Mean offset of ',
num2str(round(nanmean(nanmean(sum(abs(LowerBlur-ScaledUpperBlur),3)/80)))), ' per
pixel per band'], 'FontSize', 20)
    colorbar
    colormap(jet)
    drawnow
end

NewUpper(Upper < 1.1) = 0;
test=geotiffinfo(num2str(n+1));
geotiffwrite([num2str(n+1),'_Optimized'], int16(NewUpper), R2, 'GeoKeyDirectoryTag',
test.GeoTIFFTags.GeoKeyDirectoryTag);
disp(['Completed image ', num2str(n+1),'.'])
clear FullLowerSubset
clear FullUpperSubset
clear Lower
clear Upper
clear NewUpper
% comment these out if doing plots
clear NewUpperBlur
clear ScaledUpperBlur
clear UpperBlur
end
toc(All)
```

Program called when matching the radiance between two swaths. This calculates the difference between the two swaths totaled across the entire overlap area across all of the bands. This returned number is minimized by varying the parameters. The program is written so that the weights of an individual correction can be changed. This means that the program can be used to correct for only Track or only Cross Track for example.

An option for spatially weighting the result spatially. This could be justified based on the amount of vignette that a lens introduces, but was determined to be unnecessary with the system used.

```
function [Output] = OverlapBrightnessSimple(Master, Slave, InitialWeights, Parameters)
%Input should be a (CrossTrack, Track, Intensity) type matrix which will
%match the Output in size and format.
Dim = size(Slave);
%Break out inputs
CrossTrackWeight = InitialWeights(1);
IntensityWeight = InitialWeights(2);
CTSlope = Parameters(1);
CTIntercept = Parameters(2);
ISlope = Parameters(3);
IIntercept = Parameters(4);
Output1 = 0;
Output2 = 0;
 if CrossTrackWeight > 0 %add CrossTrack scaling
    Output1 = Slave.*repmat(CTSlope*(1:Dim(1))'+CTIntercept,1,Dim(2),Dim(3));
end
if IntensityWeight > 0 %add Intensity scaling
    Output2 = ISlope*Slave+IIntercept;
end
%add all of the contributions with weightings
ScaledInput = (CrossTrackWeight*Output1 + IntensityWeight*Output2);%.*Weighting;

Output4 = abs(Master-ScaledInput); %This will be the value to minimize
Output4(isnan(Output4)) = 0;
Output = sum(sum(sum(Output4)));
end
```

Two different implementations to interpolate data across gaps in the data on a band-by-band basis. The first is described in the thesis. The second is a simpler implementation which simply averages over large areas, this versions is useful when there are gaps in the data approximately 50 pixels or larger. After running either of these on a mosaiced image the spatial processing should be complete.

```
%% Fill in gaps in HS data
[HS0,R1] = geotiffread('hs80band1m_3doverlap3'); %read in Hyperspectral Data
Dim = size(HS0);
TempHS = double(HS0);
clear HS0
TempHS(TempHS<2)=0; %replace all '0' values with 'NaN' as this is the tag for the
interpolation

parfor_progress(Dim(3));
for band = 1:Dim(3)
   se = strel('disk',2);
   TempHS(:,:,band) = imclose(TempHS(:,:,band), se);
end
TempHS(TempHS<2)=NaN;

parfor band = 1:Dim(3)
   TempHS(:,:,band) = inpaint_nans(TempHS(:,:,band),2);
   parfor_progress;
end
parfor_progress(0);

test=geotiffinfo('hs80band1m_3doverlap3');
geotiffwrite('HS80band1m_3doverlap3_Interp', int16(TempHS), R1,
'GeoKeyDirectoryTag', test.GeoTIFFTags.GeoKeyDirectoryTag);


%% Another version, simple averaging, but works better over large areas.
[HS0,R1] = geotiffread('hs80band1m_3doverlap3'); %read in Hyperspectral Data
Dim = size(HS0);
TempHS = double(HS0);
clear HS0

numIterations = 500;
avgPrecisionSize = 10; % smaller is better, but takes longer
```

```
parfor_progress(Dim(3));
parfor band = 1:Dim(3)
   % Read in the image grayscale:
   originalImage = TempHS(:,:,band);

   % get the bad pixels where  = 0 and dilate to make sure they get everything:
   badPixels = (originalImage < 2);
   badPixels = imdilate(badPixels, ones(6));

   %# Create a big gaussian and an averaging kernel to use:
   G = fspecial('gaussian',[1 1]*100,50);
   H = fspecial('average', [1,1]*avgPrecisionSize);

   %# User a big filter to get started:
   newImage = imfilter(originalImage,G,'same');
   newImage(~badPixels) = originalImage(~badPixels);

   % Now average to
   for count = 1:numIterations
     newImage = imfilter(newImage, H, 'same');
     newImage(~badPixels) = originalImage(~badPixels);
   end
   TempHS(:,:,band) = newImage;
   parfor_progress;
end
clear newImage; clear originalImage;
parfor_progress(0);

test=geotiffinfo('hs80band1m_3doverlap3');
geotiffwrite('HS80band1m_3doverlap3_Interp', int16(TempHS), R1,
'GeoKeyDirectoryTag', test.GeoTIFFTags.GeoKeyDirectoryTag);
```

Atmospheric corrections to interpolated data. This program reads in the entire set of data and determines the most vegetated and fallow pixels based off of an NDVI measurement and used the vegetation spectra to determine the best atmospheric depth and wavelength offset to minimize absorption features. This technique is sufficient if the area matches the atmospheric profile in the ASTM standard, but fails if the H_2O column is substantially larger, or the aerosol profile is different (such as flying at a higher altitude or near a city). After this correction the data should be in units of surface reflectance. Data is saved in the same format as Landsat, a 16 bit signed integer with 0-100% corresponding to 0-10000 meaning that the smallest reflectance change is 0.01%.

Also included is a program to look at the apparent spatial distribution of the atmospheric depth and wavelength offset. This should not be used for a spatially varying correction as the model is not robust enough.

```
%% Load data
[HS0,R1] = geotiffread('HS80band1m_3doverlap2_Interp');
% [HS0,R1] = geotiffread('subset');
load('C:\Users\repasky\Box Sync\data\Overlap\Constants.mat')

%% Calculate Wavelength Offset and Atmospheric Depth

%standard NDVI
NDVI = (mean(HS0(:,:,56:58),3) - mean(HS0(:,:,35:40),3)) ./ (mean(HS0(:,:,56:58),3) +
mean(HS0(:,:,35:40),3));

NDVI(NDVI < -0.2) = -0.2; %remove extreme NDVI values
NDVI(NDVI > 1.2) = 1.2;

Fallow = double(HS0).*repmat((NDVI < prctile(NDVI(:),15)),1,1,224);
Fallow(Fallow  == 0) = NaN;
AveFallow = squeeze(nanmean(nanmean(Fallow,1),2));
clear Fallow

Vegetation = double(HS0).*repmat((NDVI > prctile(NDVI(:),85)),1,1,224);
```

```
Vegetation(Vegetation  == 0) = NaN;
AveVegetation = squeeze(nanmean(nanmean(Vegetation,1),2));
clear Vegetation


%% Calculate best fit for offset and atmospheric depth
options = optimset('MaxFunEvals', 500000,'Display', 'off', 'TolFun',1e-8);

tic
bestfval = Inf;
for T = 1:5 %do multistart minimization since it is a 'simple' problem
   [x,fval] = fminsearch(@(x)AtmosphereScaling(x, AveVegetation(1:50),
SpectralDataFull, HSWavelengths(1:50)), [(rand+1), (rand-.5)*6], options);
   if fval < bestfval
      bestfval = fval;
      bestx = x;
   end
end
toc

Offset = bestx(2);
Depth = bestx(1);

Atmosphere = SpectralDataFull(:,3) - (Depth/1.5)*(SpectralDataFull(:,3) -
SpectralDataFull(:,2));

Edges = zeros(numel(HSWavelengths)+1,1);
HalfWidth = (HSWavelengths(2:end) - HSWavelengths(1:(end-1)))/2;
HalfWidth(end+1) = HalfWidth(end); %duplicate last element
Edges(2:end) = HSWavelengths + Offset + HalfWidth;
Edges(1) = HSWavelengths(1) + Offset - HalfWidth(1); %Assuming equal widths above
and below

Wavelengths = zeros(numel(HSWavelengths),1);
for B = 1:numel(HSWavelengths)
   Wavelengths(B) = trapz(Edges(B):0.1:Edges(B+1),
interp1(SpectralDataFull(:,1),Atmosphere,Edges(B):0.1:Edges(B+1)))/(Edges(B+1) -
Edges(B));
end


figure(1)
clf(1)
hold on
```

```
plot(HSWavelengths, AveFallow./mean(Wavelengths(1:end))-.1,'r-')
plot(HSWavelengths, AveFallow./Wavelengths,'b-')
plot(HSWavelengths, smooth(AveFallow./Wavelengths,20,'loess'),'c-')
plot(HSWavelengths, Wavelengths/1e5-.15,'g-')
hold off

figure(2)
clf(2)
hold on
plot(HSWavelengths, AveVegetation./mean(Wavelengths(1:end))-.1,'r-')
plot(HSWavelengths, AveVegetation./Wavelengths,'b-')
plot(HSWavelengths, smooth(AveVegetation./Wavelengths,20,'loess'),'c-')
plot(HSWavelengths, Wavelengths/1e5-.15,'g-')
hold off

%% Save to new file
HS1 =
int16(double(HS0)./repmat(reshape(Wavelengths,1,1,size(HS0,3)),size(HS0,1),size(HS0,
2))*10000);
test=geotiffinfo('subset.tif');
geotiffwrite('subset_AtmCorr', int16(HS1), R1, 'GeoKeyDirectoryTag',
test.GeoTIFFTags.GeoKeyDirectoryTag);
HSWavelengths = HSWavelengths+Offset;
save('HSWavelengths.mat', 'HSWavelengths');
```

```
%% Look at spatial distribution of atmosphere

HS0small = double(imresize(HS0,1/5, 'lanczos3'));
% NDVIsmall = imresize(NDVI,1/50, 'lanczos3');
options = optimset('MaxFunEvals', 500000,'Display', 'off', 'TolFun',1e-8);
data = zeros(size(HS0small,1), size(HS0small,2), 2);
parfor_progress(size(HS0small,1));
tic
parfor Y = 1:size(HS0small,1)
   temp = zeros(1,size(HS0small,2),2);
   for X = 1:size(HS0small,2)
      x = [NaN,NaN];
%        if NDVIsmall(Y,X) > 0.7
         bestfval = Inf;
         for T = 1:10 %do multistart minimization since it is a 'simple' problem
            [x,fval] = fminsearch(@(x)AtmosphereScaling(x, squeeze(HS0small(Y,X,:)),
SpectralData, Wavelengths'), [(rand*0.5+1), rand*.9+0], options);
            if fval < bestfval
               bestfval = fval;
               bestx = x;
            end
         end
%        end
      temp(1,X,:) = x';
   end
   data(Y,:,:) = temp(1,:,:);
   parfor_progress;
end
parfor_progress(0);
toc
figure(1); clf(1)
subplot(2,1,1)
surface(data(:,:,1),'edgecolor','none')
colormap(jet); colorbar
caxis([0.6,1.4])
axis([1,size(HS0small,2),1,size(HS0small,1)])
subplot(2,1,2)
surface(data(:,:,2),'edgecolor','none')
colormap(jet); colorbar
% caxis([3,3.6])
axis([1,size(HS0small,2),1,size(HS0small,1)])
```

Program that spatially and spectrally matches the hyperspectral data to a LaSRC data. As mentioned before there are options to create plots that show the overlap, but this is commented out to save memory and time. The spatial portion is separated out from the spectral matching as there can be instances that give unphysical solutions to the alignment. In such a case, it can usually be solved by lowering the GrowthFactor and InitialRadius parameters. This causes the two sets of data to spatially converge slower hopefully eliminating overshoot though at the expense of time.

```
%% Match HS to Landsat, first spatially, then spectrally
[HS0,R3] = geotiffread('HS80band1m_3doverlap2_Interp_AtmCorr');
[LS1,R2] = geotiffread('ls_08_30m_largearea_sr_cc'); %Landsat image larger than HS to allow for image alignment
load('C:\Users\repasky\Box Sync\data\Overlap\Constants.mat') %constants containing data about RapidEye as well as HS Wavelengths
load('HSWavelengths.mat') %image specific Wavelengths
Dim = size(HS0);
HS0 = double(HS0);
LS1 = double(LS1);
HS0(HS0<1) = NaN;
LandSatResponse2 = [HSWavelengths',
interp1(LandSatResponse(:,1),LandSatResponse(:,2:7),HSWavelengths, 'pchip')];

options = optimset('MaxFunEvals', 50000,'Display', 'off', 'TolFun',1e-8);

resizefactor = 1/7.5; %this is (HS resolution)/(resolution to compare at)
HS0 = imresize(HS0, R1.CellExtentInWorldX*resizefactor);
LS1 = imresize(LS1, R2.CellExtentInWorldX*resizefactor); %assumes pixel size is the same in X and Y directions

LSEquiv = zeros(size(HS0,1),size(HS0,2),6); %Calculate Landsat equivalent bands from hyperspectral data
for band = 1:6
   LSEquiv(:,:,band) =
sum(shiftdim(repmat(LandSatResponse2(:,band+1),1,size(HS0,1),size(HS0,2)),1).*HS0,
3)*6.39./LandSatFWHM(band);
end

%shifts the upsampled Landsat image around to better match the downsampled HS image
```

```
[optimizer,metric] = imregconfig('Multimodal');
optimizer.MaximumIterations = 10000;
optimizer.GrowthFactor = 1.005; %smaller steps if it overshoots, but slower processing
optimizer.InitialRadius = 0.001;
RE1temp = rgb2gray(LS1(:,:,2:4)/max(max(max(LS1(:,:,2:4)))));
LSEquivtemp = rgb2gray(LSEquiv(:,:,2:4)/max(max(max(LSEquiv(:,:,2:4)))));
Init = eye(3,3);
Init(3,1) = (R2.XWorldLimits(1) - R1.XWorldLimits(1)-0)*resizefactor; %this is where
you can adjust the initial transformation, ideally if square there will be a 0,0 added
Init(3,2) = (R1.YWorldLimits(2) - R2.YWorldLimits(2)+0)*resizefactor;

figure(1)
clf(1)
imshowpair(LS1(:,:,2:4)/max(max(max(LS1(:,:,2:4)))), LSEquivtemp, 'falsecolor')

tform = imregtform(RE1temp, LSEquivtemp, 'translation', optimizer, metric,
'InitialTransformation', affine2d(Init));
LS1 = imwarp_same(LS1/max(max(max(LS1))),tform)*max(max(max(LS1)));
disp(['Shifted by (', num2str(tform.T(3,1)*1/resizefactor), ',',
num2str(tform.T(3,2)*1/resizefactor), ') meters. Expected (', ...
   num2str(Init(3,1)/resizefactor), ',', num2str(Init(3,2)/resizefactor), '), Delta Shift of (',...
   num2str(tform.T(3,1)/resizefactor - Init(3,1)/resizefactor), ',',
num2str(tform.T(3,2)/resizefactor - Init(3,2)/resizefactor),').'])
LS1 = LS1(1:size(HS0,1), 1:size(HS0,2),:); %crop Landsat image to same dimensions as
input Hyperspectral image

figure(2)
clf(2)
imshowpair(LS1(:,:,2:4)/max(max(max(LS1(:,:,2:4)))), LSEquivtemp, 'falsecolor')

% registered2 = imregister(RE1temp, LSEquivtemp, 'translation', optimizer, metric);
% figure(3)
% clf(3)
% imshowpair(registered2/max(max(max(registered2))), LSEquivtemp, 'falsecolor')
```

Program that spectrally matches the hyperspectral data to the LaSRC data on a pixel-by-pixel basis. A number of past versions are included as options. Specifically there are versions for using the panchromatic band if radiance is more useful than surface reflectance. This program is highly parallelizable and benefits greatly from more cores. Data is then saved as a *.tif file with the same 0-10000 scheme as before. This data is complete in terms of being the full area of interest and radiometrically referenced to LaSRC data.

```
%% Match LS and HS data
Xmin = 1;
Ymin = 1;
% Xmax = 100;
Xmax = size(HS0,2);
% Ymax = 25;
Ymax = size(HS0,1);
SaveFitLS = zeros(Ymax,Xmax,4);
SaveFitLS(SaveFitLS<0.5) = NaN; %Create matrix of NaN's for ease of plotting
temp = zeros(1,Xmax,4);
Response = LandSatResponse2;
tic
parfor_progress(Ymax-Ymin); %initialize
parfor Y=Ymin:Ymax
   temp = zeros(1,Xmax,4); %to make sure it works with the parfor loop, this will run multiple slices at once
   for X=Xmin:Xmax
      Landsat = double(squeeze(LS1(Y,X,1:5))); %change depending on whether or not Pan band is present
      HS = double(squeeze(HS0(Y,X,:)));

      [x,fval,exitflag,output] = fminsearch(@(x)LandsatMatching(x,Landsat,HS,Response,LandSatFWHM),[1,-100],options);
      temp(1,X,1) = x(1);
      temp(1,X,2) = x(2);
%       temp(1,X,3) = Landsat(6)/(sum(squeeze(HS*temp(1,X,1)+temp(1,X,2)).*Response(:,7))*6.39/LandSatFWHM(6)); %Store the Pan/HS value for scaling from the linearly scaled data
      temp(1,X,3) = 1; %if Pan band is unavailable and to be used
```

```
        temp(1,X,4) =
LandsatMatching(temp(1,X,1:2)*temp(1,X,3),Landsat,HS*temp(1,X,3),Response,LandS
atFWHM);

        %Use to rigorously fix Panchromatic band (or modify to any others if applicable)
%       [y,fval,exitflag,output] =
fminsearch(@(y)LandsatMatchingGainOnly(y,Landsat,HS,Response,LandSatFWHM),0,
options);
%       temp(1,X,1) = y;
%       temp(1,X,2) = (1/sum(Response(:,7)))*(Landsat(6)*LandSatFWHM(6)/6.39-
y*sum(HS.*Response(:,7)));
%       temp(1,X,3) =
Landsat(6)/(sum(squeeze(HS*temp(1,X,1)+temp(1,X,2)).*Response(:,7))*6.39/LandSatF
WHM(6));
%       temp(1,X,4) =
LandsatMatching(temp(1,X,1:2)*temp(1,X,3),Landsat,HS,Response,LandSatFWHM)

    end
    parfor_progress; %show a slice as complete
    SaveFitLS(Y,:,:) = temp(1,:,:); %save all values to a common matrix
end
parfor_progress(0); %clean up
toc
nansum(nansum(SaveFitLS(:,:,4)))
% display('Fit Completed')
% save('SaveFitMin.mat', 'SaveFit'); %Saves data to a file for later retrieval and plotting
% display('Data Saved')


%% Resize found 'image' of fit parameters and crop to correct size
SaveFitLarge = imresize(SaveFitLS, [Dim(1), Dim(2)]);

%apply correction to full HS image, this image will now be matched to Landsat
[HS0,R2] = geotiffread('west_AtmCorr_Interp'); %read in data
HS0 = double(HS0);
for band = 1:size(HS0,3)
    HS0(:,:,band) = (HS0(:,:,band).*SaveFitLarge(:,:,1) +
SaveFitLarge(:,:,2)).*SaveFitLarge(:,:,3); %apply linear change
end
% HS1 = geotiffread('fieldb'); %read in data if you need to mask anything out
% HS0 = HS0.*(HS1 > 1);
test=geotiffinfo('west_AtmCorr_Interp');
```

```
geotiffwrite('HS80band1m_3doverlap3_Interp_AtmCorr_Shifted_LS_Corrected_Surface
_Reflectance', int16(HS0), R2, 'GeoKeyDirectoryTag',
test.GeoTIFFTags.GeoKeyDirectoryTag);
```

APPENDIX B

MATLAB CODE FOR ANALYSIS, TESTING, AND PLOTTING

Program to compare random areas of fixed sizes. This was used in the verification of the technique and showed the value even for areas below the size of a single Landsat pixel.

```
%% Compare Hyperspectral to LaSRC data over random areas
Index = 0;
Neigh = 0:20;
Areas = 2000;
Difference = zeros(size(Neigh,2),Areas,5);
Difference2 = zeros(size(Neigh,2),Areas,5);
MAPE = zeros(size(Neigh,2),Areas,5);
MAPE2 = zeros(size(Neigh,2),Areas,5);
for j = Neigh
   Neighbors = j;
   Index = Index + 1;
   for Area = 1:Areas
      X = Neighbors + round(rand()*(size(HS0,2)-2*Neighbors-1-100))+51; %stay at
least 50 pixels from the edges
      Y = Neighbors + round(rand()*(size(HS0,1)-2*Neighbors-1-100))+51;

      LSEMeanRaw = squeeze(mean(mean(HS0((Y-Neighbors):(Y+Neighbors),(X-
Neighbors):(X+Neighbors),:),1),2));

      LSEMeanCorr = squeeze(mean(mean(HS1((Y-Neighbors):(Y+Neighbors),(X-
Neighbors):(X+Neighbors),:),1),2));

      LSMean = squeeze(mean(mean(LS1((Y-Neighbors):(Y+Neighbors),(X-
Neighbors):(X+Neighbors),:),1),2));

      Difference(Index,Area,:) = abs((LSEMeanCorr-LSMean)./LSMean);
      Difference2(Index,Area,:) = abs((LSEMeanRaw-LSMean)./LSMean);

      MAPE(Index,Area,:) = (LSEMeanCorr-LSMean).^2;
      MAPE2(Index,Area,:) = (LSEMeanRaw-LSMean).^2;
   end
end
figure(1)
clf(1)
subplot(1,2,2)
plot((2*Neigh+1).^2,squeeze(mean(Difference,2))*100, 'LineWidth', 2)
set(gca,'fontsize',24)
legend('Coastal Blue', 'Blue', 'Green', 'Red', 'IR')
```

```
axis([0,1700,0,35])
xlabel('Sample area size (m^2)')
ylabel('% difference')
subplot(1,2,1)
plot((2*Neigh+1).^2,squeeze(mean(Difference2,2))*100, 'LineWidth', 2)
set(gca,'fontsize',24)
% legend('Coastal Blue', 'Blue', 'Green', 'Red', 'IR')
axis([0,1700,0,35])
xlabel('Sample area size (m^2)')
ylabel('% difference')

figure(2)
clf(2)
subplot(1,2,2)
plot((2*Neigh+1).^2,squeeze(sqrt(sum(MAPE,2)/Areas))./repmat(squeeze(mean(mean(L
S1)))',Index,1), 'LineWidth', 2)
set(gca,'fontsize',24)
legend('Coastal Blue', 'Blue', 'Green', 'Red', 'IR')
axis([0,1700,0,.4])
xlabel('Sample area size (m^2)')
ylabel('CV(RMSD) between Raw HS and Landsat')
subplot(1,2,1)
plot((2*Neigh+1).^2,squeeze(sqrt(sum(MAPE2,2)/Areas))./repmat(squeeze(mean(mean(
LS1)))',Index,1), 'LineWidth', 2)
set(gca,'fontsize',24)
legend('Coastal Blue', 'Blue', 'Green', 'Red', 'IR')
axis([0,1700,0,.4])
xlabel('Sample area size (m^2)')
ylabel('CV(RMSD) between Corrected HS and Landsat')
```

Program to compare different spatial sizes by generating histograms of each different size, and then by comparing global changes between sizes. Finally, there is the program that generated the exponential plot. Note: The exponential behavior is interesting as it is plotted against edge size. If it was plotted against total area it would nearly be a line implying scaling with edge size squared, however there is an issue of communication overhead that causes the time to scale faster than quadratic.

```
%% Analysis of filtered image

temp = temp11x11;
temp(temp > 6) = 6;
temp2 = sum(temp,3);
% temp2(temp2 > 20) = 20;
figure(1)
clf(1)
surface(temp2(600:1000, 2600:3100), 'edgecolor','none')
caxis([0,54])
axis([1,500,1,400])
colormap([0,0,0;jet(15)])
colorbar
set(gca,'fontsize',24)
xlabel('Distance (m)')
ylabel('Distance (m)')

[elements1, centers1] = hist(temp2(:),0:0.5:60);

temp = temp21x21;
temp(temp > 6) = 6;
temp2 = sum(temp,3);
% temp2(temp2 > 20) = 20;
figure(2)
clf(2)
surface(temp2(600:1000, 2600:3100), 'edgecolor','none')
caxis([0,54])
axis([1,500,1,400])
colormap([0,0,0;jet(15)])
colorbar
set(gca,'fontsize',24)
xlabel('Distance (m)')
```

```
ylabel('Distance (m)')

[elements2, centers2] = hist(temp2(:),0:0.5:60);

temp = temp31x31;
temp(temp > 6) = 6;
temp2 = sum(temp,3);
% temp2(temp2 > 20) = 20;
figure(3)
clf(3)
surface(temp2(600:1000, 2600:3100), 'edgecolor','none')
caxis([0,54])
axis([1,500,1,400])
colormap([0,0,0;jet(15)])
colorbar
set(gca,'fontsize',24)
xlabel('Distance (m)')
ylabel('Distance (m)')

[elements3, centers3] = hist(temp2(:),0:0.5:60);

temp = temp41x41;
temp(temp > 6) = 6;
temp2 = sum(temp,3);
% temp2(temp2 > 20) = 20;
figure(4)
clf(4)
surface(temp2(600:1000, 2600:3100), 'edgecolor','none')
caxis([0,54])
axis([1,500,1,400])
colormap([0,0,0;jet(15)])
colorbar
set(gca,'fontsize',24)
xlabel('Distance (m)')
ylabel('Distance (m)')

[elements4, centers4] = hist(temp2(:),0:0.5:60);

temp = temp51x51;
temp(temp > 6) = 6;
temp2 = flipud(sum(temp,3));
% temp2(temp2 > 20) = 20;
figure(5)
clf(5)
```

```
surface(temp2(600:1000, 2600:3100), 'edgecolor','none')
caxis([0,54])
axis([1,500,1,400])
colormap([0,0,0;jet(15)])
colorbar
set(gca,'fontsize',24)
xlabel('Distance (m)')
ylabel('Distance (m)')

[elements5, centers5] = hist(temp2(:),0:0.5:60);

temp = temp61x61;
temp(temp > 6) = 6;
temp2 = flipud(sum(temp,3));
% temp2(temp2 > 20) = 20;
figure(6)
clf(6)
surface(temp2(600:1000, 2600:3100), 'edgecolor','none')
caxis([0,54])
axis([1,500,1,400])
colormap([0,0,0;jet(15)])
colorbar
set(gca,'fontsize',24)
xlabel('Distance (m)')
ylabel('Distance (m)')

[elements6, centers6] = hist(temp2(:),0:0.5:60);

totalpixels = sum(sum(temp2 > 0));
figure(7)
clf(7)
plot(centers1,(elements1/totalpixels*100), 'r-', ...
    centers2,(elements2/totalpixels*100), 'g-.', ...
    centers3,(elements3/totalpixels*100), 'b:', ...
    centers4,(elements4/totalpixels*100), 'c-', ...
    centers5,(elements5/totalpixels*100), 'm-.', ...
    centers6,(elements6/totalpixels*100), 'k:', 'LineWidth', 4);
set(gca,'fontsize',32)
xlabel('Total MADs')
ylabel('Percentage of Occurances')
legend('11x11', '21x21', '31x31', '41x41', '51x51', '61x61')
```

```
%% Differences
Cutoff1 = 6;
Cutoff2 = Cutoff1;
Cutoff3 = Cutoff1;
Cutoff4 = Cutoff1;
Cutoff5 = Cutoff1;
Cutoff6 = Cutoff1;

A = nansum(nansum(nansum(temp21x21.*(temp21x21 < Cutoff2) -
temp11x11.*(temp11x11 <
Cutoff1),3)))/nansum(nansum(nansum(temp11x11.*(temp11x11 < Cutoff1),3)));
B = nansum(nansum(nansum(temp31x31.*(temp31x31 < Cutoff3) -
temp21x21.*(temp21x21 <
Cutoff2),3)))/nansum(nansum(nansum(temp21x21.*(temp21x21 < Cutoff2),3)));
C = nansum(nansum(nansum(temp41x41.*(temp41x41 < Cutoff4) -
temp31x31.*(temp31x31 <
Cutoff3),3)))/nansum(nansum(nansum(temp31x31.*(temp31x31 < Cutoff3),3)));
D = nansum(nansum(nansum(temp51x51.*(temp51x51 < Cutoff5) -
temp41x41.*(temp41x41 <
Cutoff4),3)))/nansum(nansum(nansum(temp41x41.*(temp41x41 < Cutoff4),3)));
E = nansum(nansum(nansum(temp61x61.*(temp61x61 < Cutoff6) -
temp51x51.*(temp51x51 <
Cutoff5),3)))/nansum(nansum(nansum(temp51x51.*(temp51x51 < Cutoff5),3)));

figure(1)
clf(1)
plot(21:10:61,abs([A,B,C,D,E]*100),'r-*', 'LineWidth', 4, 'MarkerSize',16)
set(gca,'fontsize',32)
xlabel('Edge Size (m)')
ylabel('Relative Change (%)')
axis([11,61,0,4])
xticks([11,21,31,41,51,61])
```

```
%% Make plot of time it takes to work through a 501x501 image and fit exponential or
quadratic
% timing = [6,51.42; 11,67.84; 16,70.87; 21,105.59; 26,114.48; 31,169.32; 36,190.89;
41,258.89; 46,306.95; 51,387.97];
timing = [11,67.84; 21,105.59; 31,169.32; 41,258.89; 51,387.97];
% timing = [6,51.42; 16,70.87; 26,114.48; 36,190.89; 46,306.95];

fh = @(x,p) p(1) + p(2)*exp(-x./p(3));
errfh = @(p,x,y) sum((y(:)-fh(x(:),p)).^2);
bestval = Inf;
options = optimset('Display', 'off', 'MaxIter', 1000, 'MaxFunEvals', 50000);
for G = 1:1000
   p0 = ([rand rand rand]-0.5)*25;
   [P,fval] = fminsearch(errfh,p0,options,timing(:,1),timing(:,2));
   if fval < bestval
      bestval = fval;
      bestP = P;
   end
end

% % Quadratic Fitting, not as good as exponential
% fh2 = @(x,p) p(1) + p(2)*x + p(3)*x.^2;
% errfh2 = @(p,x,y) sum((y(:)-fh2(x(:),p)).^2);
% bestval2 = Inf;
% options = optimset('Display', 'off', 'MaxIter', 1000, 'MaxFunEvals', 50000);
% for G = 1:1000
%    p0 = ([rand rand rand]-0.5)*25;
%    [P2,fval2] = fminsearch(errfh2,p0,options,timing(:,1),timing(:,2));
%    if fval < bestval2
%       bestval2 = fval2;
%       bestP2 = P2;
%    end
% end

figure(6)
clf(6)
plot(0:55, fh(0:55,bestP), 'b-', timing(:,1),timing(:,2),'r*', 'MarkerSize',16, 'LineWidth', 4)
axis([11,61,0,400])
set(gca,'fontsize',32)
xlabel('Edge Size (m)')
ylabel('Time to Fit (s)')
legend('Data', 'Exponential Fit', 'Location', 'NorthWest')
xticks([11,21,31,41,51,61])
```

Program for plotting the year to year change. Options for removing small anomalies is give in the ‘bwareaopen’ command.

```
%% Plots...
% New2015 = ((temp2015 - temp2014) > 10);
% New2015 = bwareaopen(New2015,5,4);

New2016 = ((temp2016 - temp2015) > 10);
New2016 = bwareaopen(New2016,10,4);
New2016b = ((temp2015 - temp2016) > 10);
New2016b = bwareaopen(New2016b,10,4);

% New2016b = ((temp2016 - temp2014) > 10);
% New2016b = bwareaopen(New2016b,5,4);

% Diff2 = imresize(Diff,[1760,3331]);
% Diff2 = (Diff2 > 0.1) + (Diff2 < -0.1);

figure(1)
clf(1)
surface(2*bwareaopen((temp2016 > 20),15,4) + bwareaopen((temp2015 > 20),15,4),
'edgecolor','none')
axis([1,size(temp2016,2),1,size(temp2016,1)])
% colormap([0,0,0;flipud(jet(20))])
% colormap([0,0,0;1,0,0;0,1,0])
% colormap([0,1,0;0,0.7,0;0,0.3,0;0,0,0;0,0,0;0,0,0;0,0,0;0,0,0;0,0,0;0.3,0,0;0.7,0,0;1,0,0])
% colormap([0,0,0;1,1,1])
colormap([0,0,0;1,0,0;0,1,0;0,0,1])
caxis([0,3])
colorbar

figure(3)
clf(3)
surface(New2016 + 2*New2016b, 'edgecolor','none')
axis([1,size(temp2016,2),1,size(temp2016,1)])
% colormap([0,0,0;flipud(jet(3))])
colormap([0,0,0;1,0,0;0,1,0])
% colormap([0,1,0;0,0.7,0;0,0.3,0;0,0,0;0,0,0;0,0,0;0,0,0;0,0,0;0,0,0;0.3,0,0;0.7,0,0;1,0,0])
% colormap([0,0,0;1,1,1])
caxis([0,2])
colorbar
axis off
```

```
figure(2)
clf(2)
surface(temp2016, 'edgecolor','none')
axis([1,size(temp2015,2),1,size(temp2015,1)])
colormap([0,0,0;0,0,0;0,0,0;0,0,0;jet(7)])
colorbar
caxis([0,41.5])
axis off

%
% figure(3)
% clf(3)
% surface(temp2014, 'edgecolor','none')
% axis([1,size(temp2014,2),1,size(temp2014,1)])
% colormap(jet)
% colorbar
```

Miscellaneous programs for plots in the thesis. Brief descriptions are given before each program.

Plot spectral response curves from Landsat as well as examples from the hyperspectral camera.

```
%% Plot response curves
figure(1)
clf(1)
plot(LandSatResponse(:,1), LandSatResponse(:,2:7), 'LineWidth', 2)
hold on
plot(425:0.05:925, exp(-((425:0.05:925)-752).^2/(2*1.5^2)),'r-', 'LineWidth', 2,
'LineStyle', '-.')
plot(425:0.05:925, exp(-((425:0.05:925)-758.25).^2/(2*1.5^2)), 'g-', 'LineWidth', 2,
'LineStyle', '-.')
hold off
xlabel('Wavelength (nm)')
ylabel('Relative Spectral Response')
axis([425,1150,0.005,1])
set(gca,'fontsize',32)
l = legend('Landsat Coastal Blue', 'Landsat Blue', 'Landsat Green', 'Landsat Red', 'Landsat
IR', 'Landsat Panchromatic', 'Hyperspectral Band 1', 'Hyperspectral Band 2');
set(l, 'FontSize', 20);
```

Plot both the raw spectral data, the fit to the data and the individual pieces (Red Edge and Green Peak) of the fit.

```
%% Plot curves as well as pieces of plot
X = round(rand()*size(HS0,2));
Y = round(rand()*size(HS0,1));
% X = 2555;
% Y = 194;
% X = 4011;
% Y = 71;
% X = 288;
% Y = 1066;
figure(4)
clf(4)
plot(HSWavelengths(1:75), squeeze(HS0(Y,X,1:75)/100), 'k.', 'LineWidth', 2,
'MarkerSize', 20)
hold on
plot(HSWavelengths(1:75),FitVegetation(SaveFit(Y,X,:), HSWavelengths(1:75))/100,
'm-', 'LineWidth', 2)
plot(HSWavelengths(1:75),FitVegetation(squeeze(SaveFit(Y,X,:)).*[1,1,1,1,1,   0,0,1,0,
0]', HSWavelengths(1:75))/100, 'r-', 'LineWidth', 2, 'LineStyle', ':')
plot(HSWavelengths(1:75),FitVegetation(squeeze(SaveFit(Y,X,:)).*[1,0,1,1,1,   1,1,1,1,
0]', HSWavelengths(1:75))/100, 'g-', 'LineWidth', 2, 'LineStyle', '--')
% plot(HSWavelengths(1:75),squeeze(SaveFit(Y,X,2)) +
FitVegetation(squeeze(SaveFit(Y,X,:)).*[1,0,1,1,1,   0,0,1,0,    0]',
HSWavelengths(1:75)), 'b-', 'LineWidth', 2, 'LineStyle', '-.')
% plot(HSWavelengths(5:80),squeeze(SaveFit(Y,X,2)) +
FitVegetationOxygen(squeeze(SaveFit(Y,X,:)).*[1,0,1,1,1,   0,0,1,0,    0,1,0,  1,1,1,
0,1,0,  0]', 5:80))
% plot(HSWavelengths(5:80),squeeze(SaveFit(Y,X,2)) +
FitVegetationOxygen(squeeze(SaveFit(Y,X,:)).*[1,0,1,1,1,   0,0,1,0,    0,1,0,  0,1,0,
1,1,1,  0]', 5:80))
hold off
% title(['Location (', num2str(X), ',', num2str(Y),')'], 'FontSize', 24)
axis([425,925, 0, 50])
set(gca,'fontsize',32)
xlabel('Wavelength (nm)')
ylabel('Reflectance (%)')
```

Plot examples of the pieces of the fits to visualized how the parameters change the shape of the pieces.

```
%% Plot examples of pieces of Vegetation Fit
figure(1)
clf(1)
hold on
plot(HSWavelengths(5:80),FitVegetationOxygen([100,3500,48,.2,40,  0,0,1,0,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'r-', 'LineWidth', 2)
plot(HSWavelengths(5:80),FitVegetationOxygen([100,3500,48,.2,150,  0,0,1,0,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'g-', 'LineWidth', 2)
plot(HSWavelengths(5:80),FitVegetationOxygen([100,3500,48,1,40,  0,0,1,0,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'b-', 'LineWidth', 2)
plot(HSWavelengths(5:80),FitVegetationOxygen([250,4000,45,.2,40,  0,0,1,0,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'k-', 'LineWidth', 2)
hold off
axis([450,925,0,5000])
set(gca,'fontsize',32)
xlabel('Wavelength (nm)')
ylabel('% Reflectance (*10000)')
figure(2)
clf(2)
hold on
plot(HSWavelengths(5:80),FitVegetationOxygen([0,0,48,.2,40,  20000,4,17,1,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'r-', 'LineWidth', 2)
plot(HSWavelengths(5:80),FitVegetationOxygen([0,0,48,.2,40,  40000,6,17,1,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'g-', 'LineWidth', 2)
plot(HSWavelengths(5:80),FitVegetationOxygen([0,0,48,.2,40,  20000,4,25,1,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'b-', 'LineWidth', 2)
plot(HSWavelengths(5:80),FitVegetationOxygen([0,0,45,.2,40,  20000,4,17,.2,  0,1,0,
0,1,0,  0,1,0,  0,1,0, 0,1,0], 5:80), 'k-', 'LineWidth', 2)
hold off
axis([450,925,0,5000])
set(gca,'fontsize',32)
xlabel('Wavelength (nm)')
ylabel('% Reflectance (*10000)')
```

Plot spectra before and after correction to Landsat data.

```
%% Plot Spectrum before and after Landsat Matching
HS0 = geotiffread('HS80band1m_Interp');
RE1 = geotiffread('ls_12_1m.tif');
load('C:\Users\repasky\Desktop\data\Overlap\Constants.mat') %constans containing data
about RapidEye as well as HS Wavelengths
Dim = size(HS0);
HS0 = double(HS0);
RE1 = double(RE1);
Response = LandSatResponse2;

%rescale Landsat data to full range based on ENTIRE range of values
Minimum = min(min(RE1(RE1>1))); %find smallest non zero value
RE1 = RE1-Minimum; %scale down
RE1(RE1<0) = 0; %set negative numbers back to 0
Maximum = max(max(max(RE1))); %find maximum value
RE1 = RE1*((2^15-1)/Maximum); %scale image to 0-32767 (int8)
options = optimoptions(@fminunc, 'MaxFunEvals', 500,'Display', 'off', 'TolFun',1e-8);
resizefactor = 1/7.5; %value from 0-1, 1/8 seems to work well
HS0 = imresize(HS0, resizefactor);
RE1 = imresize(RE1, resizefactor);

X = round(rand()*size(HS0,2));
Y = round(rand()*size(HS0,1));

Landsat = double(squeeze(RE1(Y,X,1:5)));
HS = double(squeeze(HS0(Y,X,:)));

x0 = [3,50];
[x,fval,exitflag,output] = fminsearch(@(x)LandsatMatching(x,Landsat,HS,Response), x0,
options); %around 230/second
figure(1)
clf(1)
plot(1:5, Landsat, 'b-o', 1:5, sum(repmat(HS*(x(1))+x(2),1,5).*Response(:,2:6),1),'g-o',
1:5, sum(repmat(HS,1,5).*Response(:,2:6),1),'r-o')
title(['Slope, Intercept = (', num2str((x(1))), ',', num2str(x(2)),')'])
figure(3)
clf(3)
plot(1:80, HS, 1:80, HS*(x(1))+x(2))
title(['Location (', num2str(X), ',', num2str(Y),')'])
```

Comparing year to year data over the same are before and after correction to Landsat data.

```
%% Compare year to year variation within areas
Size = 500;

X1 = round(rand()*(size(HS2014before,2)-Size-1))+1; %pick upper left corner of first area
Y1 = round(rand()*(size(HS2014before,1)-Size-1))+1;
% X1 = 1640;
% Y1 = 2307;
X1 = 1128;
Y1 = 772;
X1 = 1000;
Y1 = 600;
% X2 = round(rand()*(size(HS2014before,2)-Size-1))+1; %pick upper left corner of second area
% Y2 = round(rand()*(size(HS2014before,1)-Size-1))+1;

Subset2014before = squeeze(mean(mean(double(HS2014before(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));
Subset2014after = squeeze(mean(mean(double(HS2014after(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));
% Subset2014LSTOA = squeeze(mean(mean(double(LS2014TOA(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));
Subset2014LSSR = squeeze(mean(mean(double(LS2014SR(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));

Subset2015before = squeeze(mean(mean(double(HS2015before(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));
Subset2015after = squeeze(mean(mean(double(HS2015after(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));
% Subset2015LSTOA = squeeze(mean(mean(double(LS2015TOA(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));
Subset2015LSSR = squeeze(mean(mean(double(LS2015SR(Y1:(Y1+Size), X1:(X1+Size), :)),1),2));

figure(1)
clf(1)
plot(418.07+(1:80)*6.39,Subset2014before*.01, 'b-', 418.07+(1:80)*6.39, Subset2015before*.01, 'g-', 'LineWidth', 2);
axis([425,925,0,60])
```

```
hold on
plot([443.63,481.97,555.46,654.50,862.18],Subset2014LSSR(1:5)*0.01, 'b*',
'LineWidth', 2, 'MarkerSize',8)
plot([443.63,481.97,555.46,654.50,862.18],Subset2015LSSR(1:5)*0.01, 'g*',
'LineWidth', 2, 'MarkerSize',8)

l1 = legend('HS 2014 Before Correction', 'HS 2015 Before Correction', 'Landsat 2014
Surface Reflectance', 'Landsat 2015 Surface Reflectance', 'location', 'NorthWest');
set(l1, 'FontSize', 20);
xlabel('Wavelength (nm)')
ylabel('Surface Reflectance (%)')
set(gca,'fontsize',30)

% Second year plot
figure(2)
clf(2)
plot(418.07+(1:80)*6.39,Subset2014after*.01, 'b-', 418.07+(1:80)*6.39,
Subset2015after*.01, 'g-', 'LineWidth', 2);
axis([425,925,0,60])

hold on
plot([443.63,481.97,555.46,654.50,862.18],Subset2014LSSR(1:5)*0.01, 'b*',
'LineWidth', 2, 'MarkerSize',8)
plot([443.63,481.97,555.46,654.50,862.18],Subset2015LSSR(1:5)*0.01, 'g*',
'LineWidth', 2, 'MarkerSize',8)

l1 = legend('HS 2014 After Correction', 'HS 2015 After Correction', 'Landsat 2014
Surface Reflectance', 'Landsat 2015 Surface Reflectance', 'location', 'NorthWest');
set(l1, 'FontSize', 20);
xlabel('Wavelength (nm)')
ylabel('Surface Reflectance (%)')
set(gca,'fontsize',30)
```

Example of plotting the individual values for the biophysically relevant fit parameters. This was for a 2x2 grid, but was also extended to a 3x3 grid to show all of the 9 parameters. Resizing was included for large data sets.

```
%% Make a little plot
SaveFitSmall = imresize(SaveFit,1/1);

Percentile = 5;
Axis = [Xmin,Xmax,Ymin,Ymax]/1;
figure(1)
clf(1)
subplot(2,2,1)
surface(SaveFitSmall(:,:,1),'EdgeColor','none')
title('Vegetation Brightness')
axis(Axis)
caxis([prctile(reshape(SaveFitSmall(:,:,1), 1, []), Percentile),
prctile(reshape(SaveFitSmall(:,:,1), 1, []), 100-Percentile)]);
colorbar
subplot(2,2,2)
surface(SaveFitSmall(:,:,2),'EdgeColor','none')
title('Intensity')
axis(Axis)
caxis([prctile(reshape(SaveFitSmall(:,:,2), 1, []), Percentile),
prctile(reshape(SaveFitSmall(:,:,2), 1, []), 100-Percentile)]);
colorbar
subplot(2,2,3)
surface(SaveFitSmall(:,:,3),'EdgeColor','none')
title('Rededge')
axis(Axis)
caxis([prctile(reshape(SaveFitSmall(:,:,3), 1, []), Percentile),
prctile(reshape(SaveFitSmall(:,:,3), 1, []), 100-Percentile)]);
colorbar
subplot(2,2,4)
surface(SaveFitSmall(:,:,6),'EdgeColor','none')
title('Green Peak Height')
axis(Axis)
caxis([prctile(reshape(SaveFitSmall(:,:,6), 1, []), Percentile),
prctile(reshape(SaveFitSmall(:,:,6), 1, []), 100-Percentile)]);
colorbar
colormap(jet)
```

APPENDIX C

MATLAB CODE FOR BIOPHYSICALLY RELEVANT PARAMETER FITTING

Program and associated function for fitting individual pixels with the biophysically relevant basis functions. There are a number of options that allow for better use of the computer doing the fitting as the total time can take more than 30 hours with this implementation. Options include setting a start delay and a total run time so that the fitting can be constrained to when the computer is not in use. Due the parallelizable nature of the fitting the memory overhead can be quite large meaning that the machine should be dedicated to the fitting task if possible.

Also included are a number of different min/max bounds for different data types: 80 band Pika data, 240 band Pika data, visible AVIRIS data, and full AVIRIS data. A number of different fit functions have been tried depending on the removal (or lack thereof) of the atmosphere and the spectral range of the data (Pika versus AVIRIS).

Data can be saved as a GeoTIFF if desired for display and analysis in other programs, but in natively saved as a *.mat file. Data can be left in a band based format or converted to wavelength based on the wavelength offset determined in the atmospheric correction. This is generally the most widely useful format and what is generally used for analysis.

```
%%  Read in DataCubes
% [HS0,R2] =
geotiffread('HS80band1m_3doverlap3_Interp_AtmCorr_Shifted_LS_Corrected_Surface_
Reflectance'); %the 80 band version
options = optimset('Display', 'off'); %default options for fit
RUN_TIME = 66*60*60; %time time in SECONDS to run the fitting routine, it will
always finish a Y slice so it may take longer than this

Xmin = 1;
Xmax = size(HS0,2);
```

```
Xchunk = 4; %the minimum number of X's to look at (size of vertical swath), 4 wide takes ~10 minutes, and is a good place to work with.

Ymin = 1;
Ymax = size(HS0,1);

if 0 %whether or not to load previous data (1) or just simply start a new matrix (0)
   load('SaveFitLSCorrAtmCorrSR1;3876.mat') %load preexisting matrix
   SaveFit2 = SaveFit; %Create temporary matrix of existing data
   SaveFit2(isnan(SaveFit2)) = 0; %0's out NaN values
   Dim2 = size(SaveFit2);
   SaveFit = zeros(Ymax,Xmax,18); %Create newly sized data
   SaveFit(1:Dim2(1), 1:Dim2(2), :) = SaveFit(1:Dim2(1), 1:Dim2(2), :) + SaveFit2;
%Adds in preexisting values
   clear SaveFit2 %Removes from memory

   if Xmin <= Dim2(2)
      disp('!!! You may be overwriting previous data, check Xmin dimension. !!!')
      pause(10)
   end
else
   SaveFit = zeros(Ymax,Xmax,18);%Create newly sized data
end
pause(0*60*60) % use if you want a delayed start for any reason
TIME = tic;

% % Wait until license is available
% Count = 0;
% while (~license('checkout','Distrib_Computing_Toolbox'))
%     pause(10);
%     Count = Count+1;
% end

LoopCount = 0;
XmaxInt = 0;
while (toc(TIME) < RUN_TIME) && (XmaxInt < Xmax)
   XminInt = Xmin+LoopCount*Xchunk;
   XmaxInt = XminInt + (Xchunk-1);
   if XmaxInt > Xmax
      XmaxInt = Xmax;
   end
   LoopCount = LoopCount+1;
   parfor Y=Ymin:Ymax
```

```
        temp = zeros(1,Xmax,18); %to make sure it works with the parfor loop, this will run
multiple slices at once
        for X=XminInt:XmaxInt
            Input = double(squeeze(HS0(Y,X,5:end)));
%               Brightness = X(1);
%               Intensity = X(2);
%               RedEdge = X(3);
%               Steepness = X(4);
%               Squash = X(5);
%
%               GreenPeakHeight = X(6);
%               GreenPeakWidth = X(7);
%               GreenPeakCenter = X(8); %should be very near band 16
%               Skew = X(9);
%
            Min =  [0,      0,      10, 0.001, 0.1,     0,       0.1, 5,  0   ]; %use for 80 band data
            Init = [400,    3000,   40, 0.1,   10,      20000,   5,   18, 0.05];
            Max =  [100000, 100000, 70, 2,     10000,   2000000, 20,  30, 10  ];
%               Min =  [0,      0,      30,  0.001, 0.1,     0,       0.1, 15, 0   ]; %use for 240 band
data
%               Init = [400,    3000,   120, 0.1,   10,      20000,   5,   60, 0.05];
%               Max =  [100000, 100000, 210, 2,     10000,   2000000, 20,  90, 10  ];
%               Min =  [0,      0,      6,  0.001, 0.1,     0,       0.1, 3,  0   ]; %use for AVIRIS data
%               Init = [400,    3000,   25, 0.1,   10,      20000,   5,   12, 0.05];
%               Max =  [100000, 100000, 46, 2,     10000,   2000000, 20,  20, 10  ];
%               Min =  [0,      0,      675, 0.001, 0.1,     0,       0.1, 450, 0,    -20000000, 0.1,
1500, -50,  -20000000, 0.1, 1900, -50 ]; %use for AVIRIS  data
%               Init = [1000,   3000,   725, 0.1,   10,      20000,   30,  550, 0.05, 200000,    150,
1650, 0.05, 200000,    150, 2200, 0.05];
%               Max =  [100000, 100000, 775, 2,     10000,   2000000, 120, 650, 10,
20000000,  300, 1800, 50,   20000000,  300, 2500, 50  ];

            [FitV, resnorm] = lsqcurvefit(@FitVegetation,  Init, HSWavelengths(5:end)',
Input, Min, Max, options);
            Rsquared = (1-resnorm/sum((Input - mean(Input)).^2));  %calculate R^2 value for
comparison

%               [FitV, resnorm] = lsqcurvefit(@FitVegetation,  Init, 1:75, Input',Min, Max,
options);
%               Rsquared = 1-resnorm/sum((Input - mean(Input)).^2);  %calculate R^2 value
for comparison
%
            Fit = [FitV, Rsquared]; %does about 29.4 points per second (12/9/2016)
```

```
        temp(1,X,1:18) = Fit'; %stores fit values
      end
      SaveFit(Y,:,:) = SaveFit(Y,:,:)+temp(1,:,:); %save all values to a common matrix
    end
    disp([num2str(Xmin), ' to ', num2str(XmaxInt), ' completed in ', sec2hms(toc(TIME)), '
at ',num2str((XmaxInt-Xmin+1)*Ymax/toc(TIME)), ' points/second. Estimated time
remaining is ', sec2hms((Xmax-XmaxInt-1)*Ymax/((XmaxInt-
Xmin+1)*Ymax/toc(TIME))), '.'])
end
toc(TIME)
disp(['Computed ', num2str((Ymax-Ymin+1)*(XmaxInt-Xmin+1)), ' points, at a rate of
',num2str((Ymax-Ymin+1)*(XmaxInt-Xmin+1)/toc(TIME)), ' points/second.'])
display('Fit Completed')
SaveFit(SaveFit == 0) = NaN; %Replace 0's with NaN's for ease of plotting
save('SaveFitLSCorrAtmCorrSR.mat', 'SaveFit'); %Saves data to a file for later retrieval
and plotting
display('Data Saved')
endtime = clock;
```

```
%% Convert band based fits into wavelength based fits (because of atmospheric shifting or the like)

load('SaveFitLSCorrAtmCorrSR.mat') %load preexisting matrix
load('HSWavelengths.mat') %image specific Wavelengths
P = polyfit(1:80,HSWavelengths,1);
M = P(1);
b = P(2);
SaveFit0 = SaveFit;
SaveFit2 = SaveFit.*0;

SaveFit2(:,:,1) = SaveFit(:,:,1); %Brightness doesn't change
SaveFit2(:,:,2) = SaveFit(:,:,2); %Intensity doesn't change
SaveFit2(:,:,3) = M * SaveFit(:,:,3) + b; %RedEdge changes linearly
SaveFit2(:,:,4) = SaveFit(:,:,4) / M; %Steepness scales as 1/slope
SaveFit2(:,:,5) = SaveFit(:,:,5) * M^2; %Squash scales as slope^2 based on definition in fit
SaveFit2(:,:,6) = SaveFit(:,:,6) * M; %GreenPeakHeight scales by slope, based on exGaussian definition
SaveFit2(:,:,7) = SaveFit(:,:,7) * M; %GreenPeakWidth scales by slope
SaveFit2(:,:,8) = M * SaveFit(:,:,8) + b; %GreenPeakCenter changes linearly
SaveFit2(:,:,9) = SaveFit(:,:,9) / M; %Skew scales as 1/slope
SaveFit2(:,:,10) = SaveFit(:,:,10); %R^2 doesn't change

SaveFit = SaveFit2;

save('SaveFitLSCorrAtmCorrWavelengthSR.mat', 'SaveFit'); %Saves data to a file for later retrieval and plotting
```

```
%% Save the fit data as a GeoTiff
[HS0,R2] =
geotiffread('HS80band1m_3doverlap2_Interp_AtmCorr_Shifted_LS_Corrected_Surface_
Reflectance'); %the 80 band version

CellExtentInWorldXOld = (R2.XWorldLimits(2)-R2.XWorldLimits(1))/size(HS0,2);
%X pixel size of ORIGINAL data
CellExtentInWorldYOld = (R2.YWorldLimits(2)-R2.YWorldLimits(1))/size(HS0,1);
%Y pixel size of ORIGINAL data

% CellExtentInWorldXNew = CellExtentInWorldXOld*ResizeFactor; %X pixel size of
NEW data
% CellExtentInWorldYNew = CellExtentInWorldYOld*ResizeFactor; %Y pixel size of
NEW data

R2.XWorldLimits(1) = R2.XWorldLimits(1);
R2.XWorldLimits(2) = R2.XWorldLimits(1) + (Xmax)*CellExtentInWorldXOld;

R2.YWorldLimits(2) = R2.YWorldLimits(2);
R2.YWorldLimits(1) = R2.YWorldLimits(2) - (Ymax)*CellExtentInWorldYOld;

R2 = maprasterref('RasterSize',[(Ymax), (Xmax)], 'XWorldLimits', R2.XWorldLimits,
'YWorldLimits', R2.YWorldLimits);

%Mask out areas
% SaveFit(1:550, 1:1400, :) = NaN; %mask out upper left portion of very overexposed
data
% SaveFit(1:200, 1:1650, :) = NaN;
% SaveFit(:,:,10) = [];

%Normalize
Dim = size(SaveFit);
SaveFit = 10000*(SaveFit -
repmat(nanmin(nanmin(SaveFit)),Dim(1),Dim(2)))./repmat(nanmax(nanmax(SaveFit -
repmat(nanmin(nanmin(SaveFit)),Dim(1),Dim(2)))),Dim(1),Dim(2));
SaveFit2 = SaveFit*0;
for n = 1:size(SaveFit,3)
   SaveFit2(:,:,n) = flipud(SaveFit(:,:,n));
end
test=geotiffinfo('HS80band1m_3doverlap2_Interp_AtmCorr_Shifted_LS_Corrected_Surf
ace_Reflectance');
geotiffwrite('FitParametersAtmCorrLSCorrWavelengthSRNormalized', SaveFit2, R2,
'GeoKeyDirectoryTag', test.GeoTIFFTags.GeoKeyDirectoryTag);
```

```
function [ Spectra ] = FitVegetation( X, Input )
   Brightness = X(1);
   Intensity = X(2);
   RedEdge = X(3);
   Steepness = X(4);
   Squash = X(5);

   GreenPeakHeight = X(6);
   GreenPeakWidth = X(7);
   GreenPeakCenter = X(8); %should be very near band 19
   Skew = X(9);

   %spectra is sum of atan and exponentially modified gaussian peak
   ExGaussian = GreenPeakHeight * Skew * exp( ( GreenPeakWidth * Skew )^2 / 2 - ( 
Input - GreenPeakCenter ) * Skew ) .* normcdf( ( Input - GreenPeakCenter ) / 
GreenPeakWidth - GreenPeakWidth * Skew );
   if sum(~isfinite(ExGaussian)) ~= 0
      ExGaussian = GreenPeakHeight * 1 / ( GreenPeakWidth * sqrt( 2 * pi ) ) * exp( -( 
Input - GreenPeakCenter ).^2 / ( 2 * GreenPeakWidth^2 ) );
   end

   Spectra = (Intensity*(atan((Input-RedEdge)*Steepness.*1./exp(-(Input-
RedEdge).^2/Squash))/pi+1/2)+Brightness) + ExGaussian;
end



function [ Spectra ] = FitVegetationAVIRIS( X, Input )
   Brightness = X(1);
   Intensity = X(2);
   RedEdge = X(3);
   Steepness = X(4);
   Squash = X(5);

   GreenPeakHeight = X(6);
   GreenPeakWidth = X(7);
   GreenPeakCenter = X(8);
   GreenSkew = X(9);

   NIRPeakHeight = X(10);
   NIRPeakWidth = X(11);
   NIRPeakCenter = X(12);
   NIRSkew = X(13);

   SWIRPeakHeight = X(14);
```

```
    SWIRPeakWidth = X(15);
    SWIRPeakCenter = X(16);
    SWIRSkew = X(17);

    %spectra is sum of atan and exponentially modified gaussians
    ExGaussian = GreenPeakHeight * GreenSkew * exp( ( GreenPeakWidth * GreenSkew
)^2 / 2 - ( Input - GreenPeakCenter ) * GreenSkew ) .* normcdf( ( Input -
GreenPeakCenter ) / GreenPeakWidth - GreenPeakWidth * GreenSkew );
    if sum(~isfinite(ExGaussian)) ~= 0
        ExGaussian = GreenPeakHeight * 1 / ( GreenPeakWidth * sqrt( 2 * pi ) ) * exp( -(
Input - GreenPeakCenter ).^2 / ( 2 * GreenPeakWidth^2 ) );
    end

    ExGaussian2 = NIRPeakHeight * NIRSkew * exp( ( NIRPeakWidth * NIRSkew )^2 /
2 - ( Input - NIRPeakCenter ) * NIRSkew ) .* normcdf( ( Input - NIRPeakCenter ) /
NIRPeakWidth - NIRPeakWidth * NIRSkew );
    if sum(~isfinite(ExGaussian2)) ~= 0
        ExGaussian2 = NIRPeakHeight * 1 / ( NIRPeakWidth * sqrt( 2 * pi ) ) * exp( -(
Input - NIRPeakCenter ).^2 / ( 2 * NIRPeakWidth^2 ) );
    end

    ExGaussian3 = SWIRPeakHeight * SWIRSkew * exp( ( SWIRPeakWidth *
SWIRSkew )^2 / 2 - ( Input - SWIRPeakCenter ) * SWIRSkew ) .* normcdf( ( Input -
SWIRPeakCenter ) / SWIRPeakWidth - SWIRPeakWidth * SWIRSkew );
    if sum(~isfinite(ExGaussian3)) ~= 0
        ExGaussian3 = SWIRPeakHeight * 1 / ( SWIRPeakWidth * sqrt( 2 * pi ) ) * exp( -(
Input - SWIRPeakCenter ).^2 / ( 2 * SWIRPeakWidth^2 ) );
    end

    Spectra = (Intensity*(atan((Input-RedEdge)*Steepness.*1./exp(-(Input-
RedEdge).^2/Squash))/pi+1/2)+Brightness) + ...
        (-1*Intensity*(atan((Input-1250)*Steepness.*1./exp(-(Input-
RedEdge).^2/Squash))/pi+1/2)) + ...
        ExGaussian + ExGaussian2+ ExGaussian3;
end
```

APPENDIX D

MATLAB CODE FOR CLASSIFICATION AND CLUSTER MERGING

Program used for the histogram based unsupervised classification based on biophysically relevant parameters. Some historical aspects of past versions remain. Primarily the ability to do the classification in a random order or parameters a number of different times (referred to as Trees) and then combing these classifications. This is rarely used, but was not rewritten to reflect this fact and is simply mitigated by using "Trees = 1;". Random or ordered parameters are an option though generally ordered parameters work well.

```
%% Classify based on Histogram Splitting
% Brightness = X(1);
% Intensity = X(2);
% RedEdge = X(3);
% Steepness = X(4);
% Squash = X(5);

% GreenPeakHeight = X(6);
% GreenPeakWidth = X(7);
% GreenPeakCenter = X(8);
% Skew = X(9);

PeakSpacing = 20; %too close just overrepresents peaks and valleys
PeakPercentile = 60; %decreasing generally gives more classes, at the expense of
eventually classifying noise
ValleyPercentile = 40;
Parameters = 1:9; %More Parameters means more time

% Parameters1 = [];
% for G = 1:10
%     Parameters1 = [Parameters1, Parameters(randperm(numel(Parameters)))];
% end
% Parameters = Parameters1;
% clear Parameters1

Parameters = repmat(Parameters,1,15); %More repetitions means more time as well
Parameters = [size(SaveFit,3),Parameters]; %initializes based on single used of 10

MinimumTotal = 200; %Smaller numbers lead to more classes
```

```
% 250 = 633 s for 184 classes Time scales linearly with number of classes for constant spacing & percentiles, and exponentially with minimum total
% 500 = 457 s for 130 classes
%1000 = 356 s for 117 classes
%2000 = 287 s for 75 classes
%3000 = 237 s for 67 classes
%4000 = 223 s for 62 classes
%5000 = 204 s for 57 classes

Trees = 1;

SaveTrial = zeros(Trees, numel(Parameters));
SaveRandom = zeros(Trees, numel(Parameters));
TreeCase = int16(zeros(size(SaveFit,1), size(SaveFit,2),50)); %initialize classifications
tic
for T = 1:Trees
   RandomParameters = Parameters;%(randperm(numel(Parameters))); %permute Parameters into random order
   SaveRandom(T,:) = RandomParameters;
end

UniqueTrees = 0;
for T = 1:Trees
   [ NewCase, Trial ] = HistogramClassify4(SaveFit, MinimumTotal, Parameters, SaveRandom(T,:), PeakSpacing, PeakPercentile, ValleyPercentile);
%   figure(2);clf(2)
%    surface(NewCase,'edgecolor','none')
%    drawnow
   if sum(ismember(SaveTrial,Trial,'rows')) < 1 %Save unique Trees
      UniqueTrees = UniqueTrees + 1;
      TreeCase(:,:,UniqueTrees) = int16(NewCase);
      SaveTrial(UniqueTrees,:) = Trial; %add in new Trial
   end
   disp(['Tree ', num2str(T), ' finished. Processing rate: ', num2str(toc/T), ' s/Tree. Found ', num2str(UniqueTrees), ' unique trees.'])
end
clear NewCase
TreeCase(:,:,(UniqueTrees+1):end) = []; %Remove balance of Trees that did not get used

[~, idx] = sort(squeeze(max(max(TreeCase))),'descend'); %sort trees by how many classes they have
SaveTrial = SaveTrial(idx,:);
SaveTrial(SaveTrial < 1) = [];
TreeCase = TreeCase(:,:,idx);
```

```
clear idx;

ClassPercent = zeros(size(TreeCase,3),size(TreeCase,3));
for T = 1:size(TreeCase,3)
    TreeCaseTemp = TreeCase(:,:,T);
    UniqueClasses = unique(TreeCaseTemp);
    for Class = 1:numel(UniqueClasses)
        C = UniqueClasses(Class);
        ClassPercent(T,Class) = sum(sum(TreeCaseTemp ==
C))/nansum(nansum((TreeCaseTemp ~= 1)))*100;
        TreeCaseTemp(TreeCaseTemp == C) = Class;
    end
end
TreeCase = TreeCaseTemp;
clear TreeCaseTemp; clear T; clear C; clear Class;

TreeCaseSmall = double(imresize(TreeCase,1/1,'nearest'));
figure(4)
clf(4)
for T = 1:size(TreeCase,3)
    [R,C] = SubplotFinder(size(TreeCaseSmall,3));
    subplot(R,C,T)
    surface(TreeCaseSmall(:,:,T),'edgecolor','none')
    axis([1,size(TreeCaseSmall,2),1,size(TreeCaseSmall,1)])
    colormap(jet)
    colorbar
    set(gca,'fontsize',24)
    xlabel('Distance (m)')
    ylabel('Distance (m)')
end
max(hist(SaveTrial,30))
clear T;clear C; clear R; clear TreeCaseSmall
```

```
function [ Classes, Trial ] = HistogramClassify4(SaveFit, MinimumTotal, Parameters,
RandomParameters, PeakSpacing, PeakPercentile, ValleyPercentile)
Classes = 500*ones(size(SaveFit,1), size(SaveFit,2));% - ~isnan(sum(SaveFit,3));
Trial = zeros(1,numel(Parameters));
Dim = size(SaveFit);

parfor_progress(numel(Parameters));
for Q = 1:numel(Parameters)
   P = find(Parameters == RandomParameters(Q), 1);
   PlotTemp = SaveFit(:,:,Parameters(P));
   ClassTemp = Classes;
   for C = min(min(ClassTemp)):max(max(ClassTemp))
      CaseTemp = (ClassTemp == C); %Split out the areas purely with case C
      if sum(CaseTemp(:)) > (2.1*MinimumTotal) %check to make sure case could
actually be split into 2 sections larger than the minimum
         [SaveClipLocations, SaveCenter] = HistogramSplits(PlotTemp.*CaseTemp,
MinimumTotal, PeakSpacing, PeakPercentile, ValleyPercentile); %Split out the data that
falls within this case
         Divisions = sum(SaveClipLocations>0,2)-1; %How many possible divisions
could this split create
         SaveCenter(1) = -Inf;SaveCenter(end) = Inf; %In order to better handle bounds
         PDCase = zeros(Dim(1), Dim(2));

         if Divisions > 1 %if it can't be divided don't bother looking
            DivisionTotal = zeros(1,Divisions);
            for D = 1:Divisions
               %Where does a given parameter in the image fall in the histogram?
               Low = SaveCenter(SaveClipLocations(D));
               High = SaveCenter(SaveClipLocations(D+1));
               PDCase = PDCase + (CaseTemp.*(PlotTemp > Low).*(PlotTemp <
High))*(C-(Divisions-1)+(D-1)); %Divisions numbered from Low to High
               DivisionTotal(D) = sum(sum(PDCase == (C-(Divisions-1)+(D-1))));
            end
            if (min(DivisionTotal) < 0.5) && (Parameters(P) == 10) %if there is a division
of 0 that is allowed so long as it is the 'broad' parameter, so continue testing, should only
occur when using broad class initialization
               [~,I] = min(DivisionTotal);
               DivisionTotal(I) = Inf;
            end
            if min(DivisionTotal) > MinimumTotal %If smallest division still is more than
MinimumPercentage split the Case
               Classes = Classes - (Classes < C)*(Divisions-1); %Decrease Class numbers
less than case C to make room for new cases
               Classes(Classes == C) = 0; %Remove case C (it has been split)
```

```
                Classes = Classes + PDCase;
                Trial(Q) = Parameters(P);
            end
        end
    end
  end
  parfor_progress;
end
parfor_progress(0);
Trial(Trial == 0) = []; %Remove parameters that weren't used, only return parameters that actually split the data
if numel(Parameters) > 1
  Trial(numel(Parameters)) = 0; %Pad vector to be 'correct' size, this assumes that at least one parameter was not used...
end
Classes = Classes - min(min(Classes)) + 1; %shift classes down to be 1:N
```

```
function [ SaveClipLocations, SaveCenter, SaveTotals, SaveElements] =
HistogramSplits(Data, MinimumTotal, PeakSpacing, PeakPercentile, ValleyPercentile)
%Handles a SINGLE parameter, no looping
if sum(Data(:) > 0) < 10000
   Bins = round(sum(Data(:) > 0)*5);
   if Bins < 10
      Bins = 10;
   end
else
   Bins = 1000;
end
[Elements, Centers] = hist2(Data(:), Bins);

Elements = cat(2,zeros(1,10),Elements,zeros(1,10))'; %pad with 0's, should help with
finding peaks on the edges
Centers = interp1(1:Bins, Centers, -9:(Bins+10), 'linear', 'extrap')'; %extrapolate centers
to match Elements
SaveCenter = Centers;
SaveElements = Elements;
if Bins > 120
   SElements = smooth(Elements,round(Bins/50),'lowess'); %smooth elements to aid in
peaks/valley finding
else
   SElements = Elements;
end

[~, PeakLocations] = findpeaks(SElements, 'MinPeakHeight',
prctile(SElements,PeakPercentile), 'MinPeakDistance', PeakSpacing);
% [~, id] = lastwarn; warning('off',id) %Suppress warnings
ValleyCutoff = -1*prctile(SElements,ValleyPercentile);

while ValleyCutoff == 0
   ValleyPercentile = ValleyPercentile + 5;
   ValleyCutoff = -1*prctile(SElements,ValleyPercentile);
end

[~, ValleyLocations] = findpeaks(-1*SElements, 'MinPeakHeight', ValleyCutoff,
'MinPeakDistance', PeakSpacing);
% [~, id2] = lastwarn; warning('off',id2) %Suppress warnings

ValleyLocations = cat(1,1,ValleyLocations,(Bins+20)); %make sure edges are considered
clipping/valley points
% figure(2)
% clf(2)
```

```
% plot(Centers, SElements, Centers(ValleyLocations), SElements(ValleyLocations), 'ro', Centers(PeakLocations), SElements(PeakLocations), 'g+')
Count = 1;
ClipLocations = [];
for P = 1:numel(PeakLocations)
   Peak = PeakLocations(P);
   ValleyLow = find(ValleyLocations < (Peak-1), 1, 'last');
   ValleyHigh = find(ValleyLocations > (Peak+1), 1, 'first');
   Count = Count + 1;
   if Count > 2.1
      if ValleyLocations(ValleyLow) == ClipLocations(Count-1) %Same place to clip between peaks
         ClipLocations(Count) = ValleyLocations(ValleyHigh);
      elseif ValleyLocations(ValleyLow) == ClipLocations(Count-2) %found 2 peaks inside of same valleys, don't count it as a new peak
         Count = Count-1;
      else %implies different value, so find clipping locations based on COM
         COM = sum(Elements(ClipLocations(Count-1):ValleyLocations(ValleyLow)).*(ClipLocations(Count-1):ValleyLocations(ValleyLow))')/sum(Elements(ClipLocations(Count-1):ValleyLocations(ValleyLow))); %Center of mass between valleys
         if abs(ClipLocations(Count-1) - COM) < abs(ValleyLocations(ValleyLow) - COM)
            ClipLocations(Count-1) = ValleyLocations(ValleyLow);
         end
         if isempty(ValleyLocations(ValleyHigh))
            ClipLocations(Count) = Bins+20;
         else
            ClipLocations(Count) = ValleyLocations(ValleyHigh); %add high side clip
         end
      end
   else
      ClipLocations((Count-1):Count) = [ValleyLocations(ValleyLow), ValleyLocations(ValleyHigh)]; %first clip found
   end
end
ClipLocations = cat(2,1,ClipLocations,(Bins+20));
ClipLocations = unique(ClipLocations); %make sure there aren't redundant locations

Clips = numel(ClipLocations)-1;
Totals = zeros(1,Clips);
for CL = 1:Clips
   Totals(CL) = sum(Elements(ClipLocations(CL):(ClipLocations(CL+1)-1))); %sum from lowest to 1 less than highest clip, similar to histcounts
```

```
    if CL == Clips
        Totals(CL) = Totals(CL) + Elements(ClipLocations(CL+1)); %add entire last
element
    end
end

while min(Totals) < MinimumTotal %So long as one area is below the limit...
    [~,P] = min(Totals); %which clipping area is the smallest (and below minimum)
    if ClipLocations(P) == 1;
        ClipLocations(P+1) = []; %Remove the higher bound since 1 stays
    elseif ClipLocations(P+1) == (Bins+20);
        ClipLocations(P) = []; %Remove the lower bound since 1020 stays
    else
        COM =
sum(Elements(ClipLocations(P):ClipLocations(P+1)).*(ClipLocations(P):ClipLocations(
P+1))')/sum(Elements(ClipLocations(P):ClipLocations(P+1)));
        if abs(ClipLocations(P) - COM) > abs(ClipLocations(P+1) - COM) %decice
whether COM is closer to upper bound or lower bound
            ClipLocations(P) = [];
        else
            ClipLocations(P+1) = [];
        end
    end

    %recalculate totals, could simply add together based on above...
    Clips = numel(ClipLocations)-1;
    Totals = zeros(1,Clips);
    for CL = 1:Clips
        Totals(CL) = sum(Elements(ClipLocations(CL):(ClipLocations(CL+1)-1))); %sum
from lowest to 1 less than highest clip, similar to histcounts
        if CL == Clips
            Totals(CL) = Totals(CL) + Elements(ClipLocations(CL+1)); %add entire last
element
        end
    end
end

SaveClipLocations = ClipLocations;
SaveTotals = Totals;
end
```

Set of programs dealing with broad classification, by defining broad classes based off of proximity to large peaks (dominant clusters). Automatic clustering is based off of spatial proximity and cluster similarity.

```
%% Combine classes based on proximity to large peaks
figure(2)
clf(2)
Sorted = sort(-ClassPercent);

[pks,locs] = findpeaks(smooth([0,ClassPercent,0],3,'loess'), 'MINPEAKHEIGHT', 1,
'MINPEAKDISTANCE', 1);
locs = (locs-1);
locs = round((locs(2:end) + locs(1:(end-1)))/2);
if locs(end) == numel(ClassPercent)
   locs(end) = [];
end
findpeaks(smooth(ClassPercent,3,'loess'), 'MINPEAKHEIGHT', 1,
'MINPEAKDISTANCE', 1)
hold on
% plot([0;locs;numel(ClassPercent)],[0;locs;numel(ClassPercent)]*0, 'k^',
'markerfacecolor', [1 0 0], 'markersize', 8)
% axis([1,252,-1,20])
set(gca,'fontsize',24)
xlabel('Cluster Number')
ylabel('Percent of Total')
hold off
%% Calculate Broad classifcation based on location splitting above
T=1;
TreeCaseBroad = TreeCase*0;
locs = [0;locs;numel(ClassPercent(T,:))];
% locs = 0:1:max(TreeCase(:));locs = locs';
for C = 1:(numel(locs)-1)
   TreeCaseBroad(:,:,T) = int16(TreeCaseBroad(:,:,T)) + int16(C*((TreeCase(:,:,T) >
locs(C)).*(TreeCase(:,:,T) <= locs(C+1))));
end
ClassPercent2 = zeros(1,(numel(locs)-1));
for Class = 1:(numel(locs)-1)
   ClassPercent2(Class) = sum(sum(TreeCaseBroad ==
Class))/numel(TreeCaseBroad)*100;
end

figure(3)
```

```
clf(3)
for T = 1:size(TreeCase,3)
   [R,C] = SubplotFinder(size(TreeCase,3));
   subplot(R,C,T)
   surface(TreeCaseBroad(:,:,T),'edgecolor','none')
   axis([1,size(TreeCaseBroad(:,:,T),2),1,size(TreeCaseBroad(:,:,T),1)])
   colormap(jet(double(max(max(max(TreeCaseBroad))))))
   colorbar
   set(gca,'fontsize',24)
   xlabel('Distance (m)')
   ylabel('Distance (m)')
end
```

```
%% Find Spatial Proximity between various classes
TreeCaseBroad = TreeCase;
ClassPercent2 = ClassPercent;
% offsets = [0,1;0,-1;    1,1;1,0;1,-1;   -1,1;-1,0;-1,-1];
offsets = [2,1;2,0;2,-1;   1,2;1,1;1,0;1,-1;1,-2;   0,2;0,1;0,-1;0,-2;     -1,2;-1,1;-1,0;-1,-1;-
1,-2; -2,1;-2,0;-2,-1];
Span = 2;

Gray = sum(graycomatrix(TreeCaseBroad, 'GrayLimits', [1,max(TreeCaseBroad(:))],
'NumLevels', max(TreeCaseBroad(:)), 'Offset',offsets),3);
for G = 1:size(Gray,1)
   Gray((G+0):G,G) = 0;
%     Gray((G+Span):end,G) = 0;
%     Gray(1:(G-Span),G) = 0;
end
Gray2 = log10(Gray);
Gray2(isinf(Gray2)) = 0;

figure(4)
clf(4)
surface(Gray2, 'edgecolor', 'none')
axis([1,size(Gray2,2),1,size(Gray2,1)])
colormap(jet)
colorbar

Contrast = 1e20;
while max(Contrast) > 5
   Gray = sum(graycomatrix(TreeCaseBroad, 'GrayLimits', [1,max(TreeCaseBroad(:))],
'NumLevels', max(TreeCaseBroad(:)), 'Offset',offsets),3);
   for G = 1:size(Gray,1)
      Gray((G+0):G,G) = 0;
      Gray((G+Span):end,G) = 0;
      Gray(1:(G-Span),G) = 0;
   end
%     Contrast = prctile(Gray,95)./prctile(Gray,50);
%     Contrast(isinf(Contrast)) = 1e10; %Change Infinities to some number
   Contrast = max(Gray);
   Contrast = Contrast./hist(double(TreeCaseBroad(:)), max(TreeCaseBroad(:))); %size
weighted contrast
   [M,I] = max(Contrast); %Index of the most size weighted contrast class
   M
   [~,J] = max(Gray(I,:)); %Best agreement in Ith class
   TreeCaseBroad(TreeCaseBroad == I) = J; %Change Ith class to Jth class
```

```
    TreeCaseBroad = TreeCaseBroad - int16(TreeCaseBroad > I);
end
Size = 1000;
while min(ClassPercent2) < (Size/numel(TreeCaseBroad)*100)
    Gray = sum(graycomatrix(TreeCaseBroad, 'GrayLimits', [1,max(TreeCaseBroad(:))],
'NumLevels', max(TreeCaseBroad(:)), 'Offset',offsets),3);
    for G = 1:size(Gray,1)
        Gray(G,G) = 0;
    end
    [~,I] = min(ClassPercent2); %Index of the smallest class
    [~,J] = max(Gray(I,:)); %Best agreement in Ith class
    TreeCaseBroad(TreeCaseBroad == I) = J; %Change Ith class to Jth class
    TreeCaseBroad = TreeCaseBroad - int16(TreeCaseBroad > I);

    ClassPercent2 = zeros(1,max(TreeCaseBroad(:)));
    for Class = 1:max(TreeCaseBroad(:))
        ClassPercent2(Class) = sum(sum(TreeCaseBroad ==
Class))/numel(TreeCaseBroad)*100;
    end
end
Gray2 = log10(Gray);
Gray2(isinf(Gray2)) = 0;
figure(4)
clf(4)
surface(Gray2, 'edgecolor', 'none')
axis([1,size(Gray2,2),1,size(Gray2,1)])
colormap(jet)
colorbar
figure(2)
clf(2)
surface(double(TreeCaseBroad),'edgecolor','none')
axis([1,size(TreeCaseBroad,2),1,size(TreeCaseBroad,1)])
colormap([1,1,1;jet(double(max(TreeCaseBroad(:))))])
colorbar
set(gca,'fontsize',24)
xlabel('Distance (m)')
ylabel('Distance (m)')
```

APPENDIX E

MATLAB CODE FOR ANOMALY DETERMINATION

Two different approach were taken in regards to determining anomalies. One was based off of creating broad clusters (much like management areas) and determining anomalies from the classes contained within these broad clusters. This is the first set of programs that could use further development. The second means of anomaly detection was described in this thesis and was based off of local (spatial) anomaly detection. The program is much simpler as well.

```
%% Different attempt at anomaly detection, need to run this to collect data about the
classes
TreeCaseBroad(isnan(TreeCaseBroad)) = 0;TreeCaseBroad = double(TreeCaseBroad);
TreeCase(isnan(TreeCase)) = 0;TreeCase = double(TreeCase);

%Calculate how large each class in within a broad class
SubClassesHist = zeros(max(TreeCaseBroad(:)),max(TreeCase(:))+1);
SubClassPercentsHist = zeros(max(TreeCaseBroad(:)),max(TreeCase(:)));
for Class = 1:max(TreeCaseBroad(:))
    TempPlot = (TreeCaseBroad == Class).*TreeCase;
    temp = unique(TempPlot);
    temp(max(TreeCase(:))+1) = 0;
    SubClassesHist(Class,:) = temp;
    SubClassPercentsHist(Class,:) = histcounts(TempPlot(:),
1:(1+max(TreeCase(:))))/sum(sum(TempPlot>0))*100;
end
clear TempPlot

tic
[~,I] = max(SubClassPercentsHist,[],2); %The largest single class in the broad
classifications
LargestMeanHist = zeros(max(TreeCaseBroad(:)), size(SaveFit,3));
LargestMedianHist = zeros(max(TreeCaseBroad(:)), size(SaveFit,3));
LargestMADHist = zeros(max(TreeCaseBroad(:)), size(SaveFit,3));
LargestStdHist = zeros(max(TreeCaseBroad(:)), size(SaveFit,3));
for C = 1:max(TreeCaseBroad(:))
    Class = I(C);
    temp = reshape(repmat(double(TreeCase ==
Class),1,1,size(SaveFit,3)).*SaveFit,numel(TreeCase),10);
    temp(temp == 0) = NaN;
    LargestMeanHist(C,:) = nanmean(temp);
    LargestMedianHist(C,:) = nanmedian(temp);
```

```
    LargestMADHist(C,:) = mad(temp,1);
    LargestStdHist(C,:) = nanstd(temp);
end
clear temp
toc


%% Where to enough parameters become anomalous, is a significantly different map
depending on Median/Mean and Std/MAD
NumParam = 3; %seems to be the best at 3 parameters
StdDevs = 20:-0.25:0.25;

Anomaly = int8(zeros(size(TreeCase,1),size(TreeCase,2),numel(StdDevs)));
parfor_progress(numel(StdDevs));
tic
for S = 1:numel(StdDevs)
    StdDev = StdDevs(S);
    AnomalyR = TreeCaseBroad.*0;
    AnomalyG = TreeCaseBroad.*0;
    for Class = 1:max(TreeCaseBroad(:))
        for Param = 1:5
            AnomalyHigh = ((LargestMedianHist(Class,Param) +
StdDev*LargestMADHist(Class,Param)) < (SaveFit(:,:,Param))).*(TreeCaseBroad ==
Class);
            AnomalyLow  = ((LargestMedianHist(Class,Param) -
StdDev*LargestMADHist(Class,Param)) > (SaveFit(:,:,Param))).*(TreeCaseBroad ==
Class);
            AnomalyR = AnomalyR + AnomalyHigh + AnomalyLow;
        end
        for Param = 6:9
            AnomalyHigh = ((LargestMedianHist(Class,Param) +
StdDev*LargestMADHist(Class,Param)) < (SaveFit(:,:,Param))).*(TreeCaseBroad ==
Class);
            AnomalyLow  = ((LargestMedianHist(Class,Param) -
StdDev*LargestMADHist(Class,Param)) > (SaveFit(:,:,Param))).*(TreeCaseBroad ==
Class);
            AnomalyG = AnomalyG + AnomalyHigh + AnomalyLow;
        end
        clear AnomalyHigh
        clear AnomalyLow
    end
    Anomaly(:,:,S) = int8(AnomalyR+AnomalyG);
    parfor_progress;
end
parfor_progress(0);
```

```
toc
clear AnomalyR
clear AnomalyG

Anomaly(Anomaly < NumParam) = 0;
Anomaly(Anomaly >=NumParam) = 1;
StdDevs = [0,fliplr(StdDevs)];


figure(6)
clf(6)
surface(StdDevs(sum(Anomaly,3)+1).*(TreeCaseBroad)./(TreeCaseBroad),'edgecolor','n
one');
colorbar
colormap([0,0,0;jet(numel(StdDevs))])
caxis([0.25,5])
axis([1,size(Anomaly,2), 1, size(Anomaly,1)])
set(gca,'fontsize',24)
xlabel('Distance (m)')
ylabel('Distance (m)')

AnomalySaveHist = StdDevs(sum(Anomaly,3)+1);

% figure(7)
% clf(7)
% surface((TreeCase == I(1)) + (TreeCase == I(2))*2 + (TreeCase == I(3))*3 +
(TreeCase == I(4))*4 + (TreeCase == I(5))*5,'edgecolor','none')
% colorbar
% colormap([0,0,0;prism(max(TreeCaseBroad(:))+1)])
% axis([1,size(Anomaly,2), 1, size(Anomaly,1)])
% set(gca,'fontsize',24)
% xlabel('Distance (m)')
% ylabel('Distance (m)')
```

```
fun = @(x) DeviationFinder(x(:));
clear temp
temp = SaveFit.*0;
tic
parfor_progress(size(SaveFit,3));
parfor Band = 1:size(SaveFit,3)
   temp(:,:,Band) = nlfilter(SaveFit(:,:,Band), [51,51], fun);
   parfor_progress;
end
parfor_progress(0);
toc


function output = DeviationFinder(PixelsInBlock)
temp = PixelsInBlock(:);

BlockMedian = nanmedian(temp);
BlockMAD = mad(temp,1);

CenterValue = temp(round(numel(temp)/2));
output = abs(BlockMedian - CenterValue)/BlockMAD;
end
```