

Cameron Penne

HYPSO-1 Change Detection with Independent Component Analysis and Time Series Hyperspectral Data

Master's thesis in Electronic Systems Design

Supervisor: Milica Orlandić

Co-supervisor: Joseph Garrett

July 2023

Cameron Penne

HYPISO-1 Change Detection with Independent Component Analysis and Time Series Hyperspectral Data

Master's thesis in Electronic Systems Design
Supervisor: Milica Orlandić
Co-supervisor: Joseph Garrett
July 2023

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electronic Systems



Norwegian University of
Science and Technology

Contents

1	Introduction	2
2	Background	4
2.1	Remote Sensing	4
2.1.1	Overview	4
2.1.2	Hyperspectral Remote Sensing	4
2.1.3	Multispectral Remote Sensing	5
2.1.4	Satellite Remote Sensing	6
2.2	HYPSON-1 Mission	7
2.2.1	Overview	7
2.2.2	Hyperspectral Imager (HSI)	8
2.2.3	Capabilities and Advantages	9
2.3	Harmful Algal Blooms (HABs)	10
2.3.1	Overview	10
2.3.2	Remote Sensing Detection of HABs	11
2.4	Anomaly Detection	12
2.5	Independent Component Analysis (ICA)	13
2.5.1	Overview	13
2.5.2	Formulation	16
2.5.3	Preprocessing	18
2.5.4	Initialization	18
2.5.5	Limitations	18
2.5.6	FastICA	19
2.6	Dimensionality Reduction	19
2.6.1	Overview	19
2.6.2	ICA-DR	20
2.6.3	Principal Component Analysis (PCA)	20
2.6.4	Automatic Target Generation Process (ATGP)	21
2.7	Change Detection	22
2.7.1	Overview	22
2.7.2	Challenges	22
2.7.3	Binary and Multiclass Change Detection	23
2.7.4	Supervised and Unsupervised Change Detection	24
2.7.5	Bitemporal and Multitemporal Change Detection	24

2.7.6	Preprocessing	25
2.7.7	Change Detection Categories	25
2.7.8	Change Detection Methods	29
2.8	Sustainable Development Goals	30
2.8.1	SDG 6: Clean Water and Sanitation	31
2.8.2	SDG 9: Industry, Innovation, and Infrastructure	31
2.8.3	SDG 13: Climate Action	31
2.8.4	SDG 14: Life Below Water	32
2.8.5	SDG 17: Partnership for the Goals	32
3	Implementation	33
3.1	Overview	33
3.2	Software Environment	33
3.3	HYPSON-1 Data	34
3.3.1	Overview	34
3.3.2	Observation Times	35
3.3.3	Lake Erie	35
3.3.4	Salish Sea	38
3.3.5	Danube River Delta	40
3.3.6	Frohavet	43
3.4	Time Series Processing	45
3.4.1	Overview	45
3.4.2	Loading Data	45
3.4.3	Calibration	46
3.4.4	Georeferencing	46
3.4.5	Resampling and Interpolation	49
3.4.6	Land Mask	53
3.4.7	Cloud Mask	55
3.4.8	Writing to NetCDF	55
3.4.9	Input File	55
3.5	Independent Component Analysis	57
3.5.1	Overview	57
3.5.2	ICA Preprocessing	57
3.5.3	ICA Initialization	59
3.5.4	ICA Data Transformation	60
3.5.5	ICA Postprocessing	61
3.6	Change Detection	62
3.6.1	Overview	62
3.6.2	Relation to ICA	62
3.6.3	Properties	63
3.6.4	Reading Data	64
3.6.5	Change Detection Techniques	65
3.6.6	Spatial Stacking	65

3.6.7 Spectral Stacking	68
3.6.8 Spectral Differencing	71
3.6.9 Spectral Differencing and Stacking Hybrid	73
3.6.10 Component Matching	76
3.7 Validation Data	83
3.7.1 NOAA HAB Forecast Validation Data	83
3.7.2 Sentinel-3 OLCI Validation Data	85
4 Results	89
4.1 Overview	89
4.2 Interpretation of Change Detection Plots	90
4.3 Lake Erie Change Detection Results	92
4.4 Salish Sea Change Detection Results	103
4.5 Danube River Delta Change Detection Results	115
4.6 Frohavet Change Detection Results	126
5 Discussion	134
5.1 Overview	134
5.2 Change Detection Techniques Comparison	134
5.2.1 Comparison to Sentinel-3 OLCI Validation Data	134
5.2.2 Comparison to NOAA HAB Forecast	142
5.2.3 Number of Meaningful Change Maps	145
5.2.4 PCC Score Distribution	148
5.2.5 Interpretability	152
5.2.6 Multitemporal Extendibility	155
5.3 Change Detection System	157
5.3.1 Anomaly Identification	157
5.3.2 Automation	158
5.4 Improvements to ICA	158
5.4.1 Algorithm	158
5.4.2 Capabilities	159
5.5 Improvements to Time Series Datasets	159
5.5.1 Georeferencing	159
5.5.2 Cloud Mask	160
5.5.3 Land Mask	160
5.5.4 Resampling	160
5.5.5 Data Format	161
5.5.6 Other	161
6 Conclusion	162
6.1 Summary	162
6.2 Change Detection Results	163
6.3 Change Detection Techniques	163
6.4 Time Series Datasets	164

- 6.5 HAB Detection System 164
- 6.6 Final Reflections 165
- 7 Glossary** **166**
- References** **168**
- A Input File** **176**
 - A.1 Processing Settings Section 176
 - A.2 File Processing Queue Section 180
 - A.3 Input File Layout 182

Chapter 1

Introduction

Change detection is a important task in remote sensing, aimed at identifying and characterizing changes on Earth's surface. It plays a central role in various applications, including land use monitoring, agriculture, forestry, and urban planning. In offshore area, change detection can also be used to monitor for changes in bodies of water such as oceans, lakes, and fjords. Many traditional change detection techniques focus on binary change detection, which are limited to determining whether a change has occurred or not. However, there is a growing need to develop more advanced methods that can distinguish between different types of changes and relay information about detected changes, enabling multiclass change detection.

The need for better change detection methods is driven in part by the growing availability of high quality and frequent Earth observation data from multispectral and hyperspectral satellites. One of these satellites, the Hyper-Spectral Small Satellite for Ocean Observation or HYPSSO-1, has been developed by the Small Satellite Lab at the Norwegian University of Science and Technology (NTNU). The objectives of HYPSSO-1 include monitoring coastal waters and to identify and track harmful algal blooms (HABs) that threaten aquaculture. Since becoming operational in 2022, HYPSSO-1 has taken many repeat observations of ocean and land targets using an onboard hyperspectral imager (HSI). The HSI allows images to be taken with at extremely high spectral resolution. Furthermore, HYPSSO-1's sun synchronous orbit and pointing capabilities have also enabled short target revisit times allowing for frequent captures of the same locations compared to other satellites. [1] The unique capabilities of HYPSSO-1 have lead to a growing dataset of hyperspectral images, many of which can be combined into time series datasets for change detection analysis. Due to the high spectral resolution of these datasets, novel and efficient change detection techniques are needed that can evolving phenomena present in bodies of water.

In this paper, several approaches that combine the anomaly detection capabilities of Independent Component Analysis (ICA) with traditional change detection techniques are proposed for analyzing hyperspectral imagery. ICA is a data transformation-based method used in signal processing that is capable of separating mixed data into statistically independent components or features. In hyperspectral images, ICA enables efficient spectral un-mixing and extraction of features that can be used to augment traditional change detection methods. [2] [3]. The ICA-based change detection approaches aim to provide multiclass, bitemporal, and feature-based change detection capabilities. ICA has several advantages that makes it a suitable algorithm for use in change detection. Firstly, it has anomaly detection capabilities and can identify and localize abnormal spectral signatures or feature

in the ocean, distinguishing them from the normal background spectral signatures. The anomalous features can indicate algal blooms or other evolving abnormal ocean phenomena which may potentially experience changes over time. Another advantage of ICA is that it is unsupervised, eliminating the need for labelled training data or prior knowledge of the features present in the hyperspectral images. Finally, ICA can recover spectral information about the data in the form of ICA weights. This information can be used to identify and characterize targets observed with ICA methods.

The development of ICA-based change detection techniques will contribute a valuable proof of concept demonstration of tracking HABs and other ocean phenomena by location and over short term periods of time. In the future, the work could be further developed to extract more specific information about detected anomalies and changes such as algae species and concentration. Additionally, the work provides a foundation that could be evolved into an automated change detection system. Information from ICA-based change detection could be used to track and predict HABs, protecting aquaculture assets and monitoring water quality in both oceans and lakes. ICA-based change detection could also be used to promote and improve sustainability within these application areas in the future. Finally, work completed on developing ICA-based change detection methods will also expand expertise in the utilization of hyperspectral data from current and future small satellite missions at NTNU. The knowledge gained in this area could in the future be used to contribute to joint ocean observation systems using a combination of satellite, aerial, and aquatic monitoring agents.

Chapter 2

Background

2.1 Remote Sensing

2.1.1 Overview

Remote sensing is the practice of acquiring information about objects or phenomena from a distance, without physical contact. It involves the use of remote sensors and instruments on satellites, aircraft, or ground-based platforms to collect data from an object or target. Remote sensing techniques capture and measure electromagnetic radiation reflected or emitted by the target. Depending on the source of the electromagnetic radiation, remote sensing techniques can either be active and passive. Passive remote sensing relies on light or other forms of electromagnetic radiation being produced by an external source such as the sun and passively received by a sensor. Passive techniques include RGB imaging, multispectral imaging, and hyperspectral imaging. The other class of remote sensing techniques, active techniques, use sensors that emit and bounce radiation off of targets to make measurements. Synthetic aperture radar (SAR) is on such example of an active remote sensing technique.

2.1.2 Hyperspectral Remote Sensing

Hyperspectral imaging, as known as imaging spectroscopy, is a remote sensing technique that is used for passive optical remote sensing. It involves capturing images with continuous spectral measurements across a range of the electromagnetic spectrum. Typically, hyperspectral imaging operates within the visible and near infrared (NIR) regions of the electromagnetic spectrum. The continuous spectral measurements gives hyperspectral images a high spectral resolution. The high spectral resolution of hyperspectral images enables the precise measurement of the spectral signature of a pixel or target. Using spectral signatures, objects or regions of materials can be characterized and accurately identified.

The spectral signatures data produced by hyperspectral imaging are generated as sets of two dimensional spectral images that contain information about how the target reflects light at a particular wavelength within the measured electromagnetic spectrum. When combined, the two dimensional spectral images form a data structure with three dimensions. These dimensions are equal to the image height, width, and number of spectral bands or channels [4]. This type of data structure is known as a datacube and is depicted in Figure 2.1.

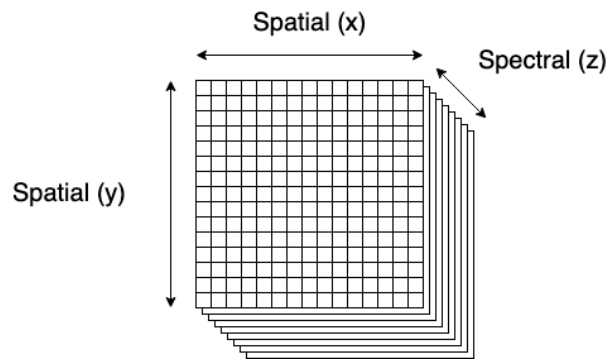


Figure 2.1: Spatial and spectral dimensions of a hyperspectral datacube.

Each spectral image within the datacube consists of exactly one spectral band. Together with the other spectral images, the datacube serves as a spectral fingerprint for targets within the hyperspectral image. By analyzing the entirety of the hyperspectral data, it is possible to obtain information describing the imaging target's composition. Since hyperspectral images can be composed of spectral signals from many different materials, the measurement of spectral signatures through hyperspectral imaging involves many aspects of classification and anomaly detection topics.

Hyperspectral datacubes contain detailed spatial and spectral information, however they require significant amounts of processing to manipulate and interpret the data [2]. In particular, simply handling the high dimensionality of the data is a challenge since the amount of data represented in a datacube can be many times that of a conventional multispectral or RGB image.

2.1.3 Multispectral Remote Sensing

Hyperspectral imaging is often contrasted to multispectral imaging, an imaging technique that does not provide as high of a spectral resolution as hyperspectral imaging. Spectral signatures from multispectral imaging are much more coarse and discontinuous and provide less overall information about imaging targets. Compared to hyperspectral imaging, multispectral imaging only samples a handful of wavelengths in the electromagnetic spectrum at discontinuous spaced intervals. Figure 2.2 offers a visual comparison between different imaging techniques. The wavelength bands that are measured in multispectral imaging are usually selected based on their usefulness for making observations or alignment with atmospheric optical windows. Like hyperspectral imaging, the bands typically range from the visible and infrared regions of the electromagnetic spectrum. Since multispectral imaging uses fewer spectral bands, it also produces less overall data. Consequently, multispectral imaging does not face as many challenges as hyperspectral imaging when it comes to efficiently storing, processing, and manipulating image data.

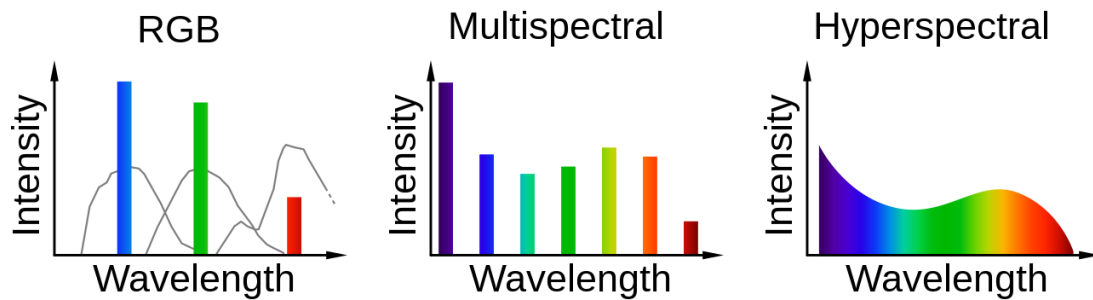


Figure 2.2: Comparison between the amount of spectral sampling in RGB, multispectral, and hyperspectral imaging [5].

Many previous studies about anomaly detection and change detection have used multispectral data rather than hyperspectral data. This is because hyperspectral imaging satellites are a relatively recent addition to satellite remote sensing compared to multispectral satellites.

2.1.4 Satellite Remote Sensing

As mentioned previously, remote sensing instruments can be carried onboard different platforms. One type of platform are orbiting satellites. Satellite remote sensing allows the Earth's surface and atmosphere from space. It involves the use of satellites equipped with passive or active imaging sensors to capture information about the Earth. It is frequently used to observe the Earth's climate and environmental conditions on land or in water. Satellite remote sensing offers several advantages over other ground-based or airborne data remote sensing methods. Firstly, satellites allow for the observation of large areas continuously as well as observation of remote or inaccessible regions. This removes the need to be physically present in a location to take measurements. Additionally, satellite data can be captured repeatedly over time at the same location, enabling the analysis of changes and trends. This is a unique advantage not possessed by airborne remote sensing instruments, which cannot remain airborne indefinitely.

An important factor for remote sensing satellites is effective revisit time. Effective revisit time is the length of time between subsequent observations of the same location. Revisit time is an especially important measure for satellite operations and scientific observations if the satellite is used to track changes or phenomena over a period of time [6]. To maximize the frequency of observation and shorten revisit times, some Earth observation satellites are placed into sun-synchronous polar orbits. Polar orbits are designed in such a way that imaging satellites pass over the Earth's poles during each orbit. This configuration allows satellites to have visibility of the entire surface of the Earth each day, as they traverse along these polar paths. Moreover, sun-synchronous orbits are configured such that the satellite pass over a location during its local noon. This means that the lighting conditions will be similar for all captures, disregarding the effects of latitude or seasonal changes.

Remote sensing satellites have carried both multispectral and hyperspectral instruments. Initially, hyperspectral remote sensing was primarily conducted through airborne imaging campaigns, where sensors were carried onboard aircraft rather than satellite platforms. This began to change beginning in the 2000s. The Earth Observation-1 (EO-1) mission by NASA's Goddard Space Flight Center (GSFC), also known as Hyperion, was significant since it was the first hyperspectral imaging satellite mission with publicly accessible hyperspectral data. Hyperion launched in 2000 and operated from

2003 until its decommissioning in 2017. [7] [8]

In recent years, there has been an increase in the number of hyperspectral satellite missions, both in the institutional and private sectors. The Italian PRISMA (PRecursores IperSpettrale della Missione Applicativa or Hyperspectral Precursor and Application Mission) satellite was launched in 2019 and is in current operation monitoring land and water environments [9]. PRISMA observes in wavelengths from 402 nm to 2371 nm and has a total of 242 spectral bands [8]. Other countries like China have also launched and operated hyperspectral satellites. GaoFen-5, an environmental monitoring satellite, was launched in 2018 [10]. Recently, independent companies have sought to develop their own hyperspectral satellites to sell and deliver data products to customers. These companies include Maxar and Orbital Sidekick [11] [12].

Historically, the availability of multitemporal hyperspectral imagery has been limited due to the scarcity of Earth observation satellites carrying hyperspectral imagers onboard as compared to multispectral imagers. Additionally, hyperspectral satellite imagery has had long revisit times for the same location, owing to the limited number of hyperspectral satellites and the narrow spatial coverage that the sensors provide. However, this landscape is changing with the emergence of current and upcoming satellite missions with new capabilities such as more frequent revisit and observation times. With these new satellites, the amount of available satellite-based hyperspectral imagery datasets will expand. As a result, the number of available multitemporal datasets is increasing significantly, creating a need for the development of efficient processing and change detection methods to effectively analyze and draw meaningful information from these datasets [4]. The ability for hyperspectral spectral satellites to make more frequent and repeated observations is also driving a need for improved hyperspectral data analysis techniques such as better change detection algorithms [8].

2.2 HYPSO-1 Mission



Figure 2.3: HYPSO-1 mission logo.

2.2.1 Overview

The Hyper-Spectral Small Satellite for Ocean Observation, or HYPSO-1, is a 6U cube satellite developed and operated by the Norwegian University of Science and Technology (NTNU) Small Satellite Lab to demonstrate the Earth observation abilities of small satellites, particularly for ocean and coastal waters monitoring. The mission objectives of HYPSO-1 are to observe the Earth's oceans with a hyperspectral imager (HSI) and to identify and track harmful algal blooms (HABs). Launched in

January 2022, HYPSON-1 is located in a 500 km altitude sun-synchronous orbit (SSO) that ensures consistent sunlight conditions during hyperspectral observations in addition to viewing access to the entirety of the Earth's surface. The operation of HYPSON-1 involves using the onboard HSI to scan swaths the Earth to generate hyperspectral images as shown in Figure 2.4. The scans of the Earth, known as hyperspectral captures, are generated as HYPSON-1 passes over the target it is observing. The hyperspectral captures can be processed onboard the satellite or saved to memory and down-linked to ground stations when the satellite comes within range of a ground antenna. [13] [1]

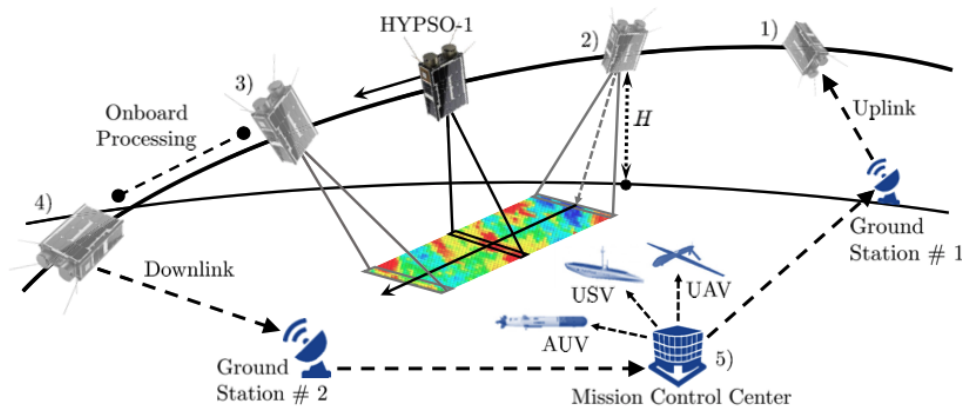


Figure 2.4: HYPSON-1 mission concept. To generate hyperspectral images, the satellite relies on its forward motion to image the Earth's surface [13].

2.2.2 Hyperspectral Imager (HSI)

The main payload of HYPSON-1 is the hyperspectral imager (HSI). The HSI was developed at NTNU and can be used to observe swaths of the Earth surface at visible wavelengths from approximately 400 nm to 800 nm [13]. It enables HYPSON-1 to produce hyperspectral images using wavelengths from a contiguous range of narrow spectral bands. This in turn enables HYPSON-1 to provide a high spectral imaging resolution for observing targets. [14]

The HSI utilizes a specific imaging technique called push-broom scanning. The push-broom scanning technique operates by obtaining an image through a narrow slit and optical prism assembly. The prism divides light into different contiguous wavelengths that are collected using a line of detectors set perpendicular to the direction of travel of the HSI and satellite. With this setup, the HSI scans the ground one pixel line at a time and relies on the satellite's forward movement to construct a full image as it passes over the target, as shown in Figure 2.5. [13]

The hyperspectral images that are produced using the HSI onboard HYPSON-1 have a resolution of 956 pixels by 684 pixels and with 120 spectral bands [13]. The first dimension (956) represents pixels in the along track direction of the capture while the second dimension (684) represents pixels in the cross track direction. These images are stored as hyperspectral datacubes, as shown in Figure 2.6. Hyperspectral datacubes are three dimensional with the first two dimensions corresponding to the height and width of the image and the third dimension corresponding to the spectral bands.

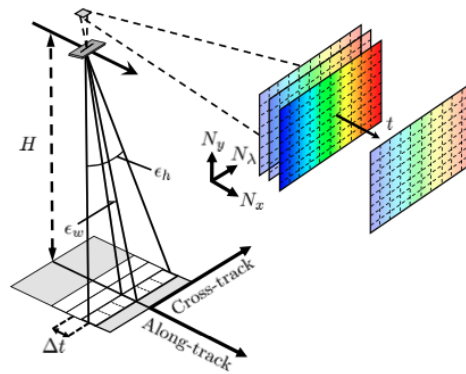


Figure 2.5: Illustration of the push-broom imaging technique. The HSI scans the ground through a small slit one line at a time in the along-track direction [13].

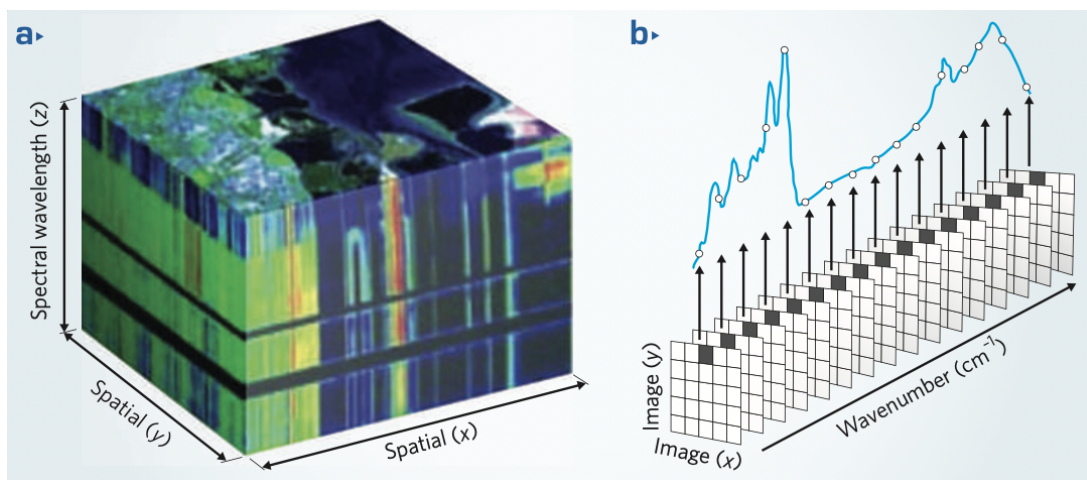


Figure 2.6: Hyperspectral datacube displaying the relationship between the spatial dimensions and spectral dimension.

2.2.3 Capabilities and Advantages

HYPSON-1 has several unique design and operational advantages that set it apart from larger satellites. Firstly, being a cube satellite, HYPSON-1 offers a cost-effective approach to its development and construction, making it a more economical option compared to larger satellites. Naturally, this puts similar satellite missions within reach for institutions and companies that are interested in the operation benefits of an Earth observing satellite.

Another advantage of HYPSON-1 is its ability to perform repeated observations of the same locations. This ability is achieved through the satellite's sun-synchronous polar orbit and pointing capabilities, which allows for controlled imaging of targets using the HSI. The pointing capability allows HYPSON-1 to rotate and orient itself towards imaging targets, allowing for hyperspectral captures to be taken even when the target is not directly below HYPSON-1. This built-in operational flexibility enables HYPSON-1 to achieve shorter revisit times and to re-observe targets within a short time frame. The revisit ability is particularly valuable for change detection purposes since it permits the creation of time series datasets over relatively brief periods, on the time span of days or weeks rather than months or years. [13] [1]

Finally, the NTNU SmallSat Lab, which operates HYPSON-1, is also able to schedule captures on

demand. In general, capture requests can be made with a lead time of a few days, depending on weather conditions and cloud cover considerations for the specific target of interest.

2.3 Harmful Algal Blooms (HABs)

2.3.1 Overview

One of the mission objectives of HYPSO-1 is to identify and track Harmful Algal Blooms, also known as HABs. HABs are large occurrences of algae that produce natural toxins which can appear suddenly in a body of water. These toxins are detrimental to the surrounding environment and ecosystems and can negatively impact aspects related to human activities such as aquaculture or drinking water quality. A common example of toxic algae is blue-green algae or cyanobacteria which can produce HABs in both oceans and lakes. [13]



Figure 2.7: Surface level image of a harmful algal bloom in Lake Erie in 2018 [15].

In recent years, there have been several instances of HAB-related events. One notable occurrence took place in 2016 when the fjords in the western Patagonia region of Chile were heavily impacted by a major algal bloom. This particular HAB was caused by a species of toxin-producing algae known as *Pseudochattonella*. Chile, being a significant producer of farm-raised salmon, experienced significant aquaculture and environmental losses as a result of this HAB. The occurrence of the 2016 bloom was linked to the El Niño–Southern Oscillation, a climatic phenomenon, which is anticipated to happen again in 2023 [16] [17]. Similarly, in 2019, Northern Norway witnessed an outbreak of toxic *Chrysochromulina* algae that led to a mass die-off of farm-produced salmon along its coasts. Much like the HAB in Chile, this event had detrimental effects on marine ecosystems and the aquaculture industry in the region. [18]

HABs are not only limited to marine environments and they can also occur in freshwater bodies. One notable example is Lake Erie, situated on the border between Canada and the United States. Lake Erie frequently experiences seasonal HABs that disrupt the quality of drinking water for residents of the region. Consequently, the seasonal monitoring of the lake and algal blooms is necessary [19]. In the case of Lake Erie, a large contributing factor to the algal bloom is agricultural run-off. The occurrence of HABs are also influenced by climate and aquatic conditions. [20].

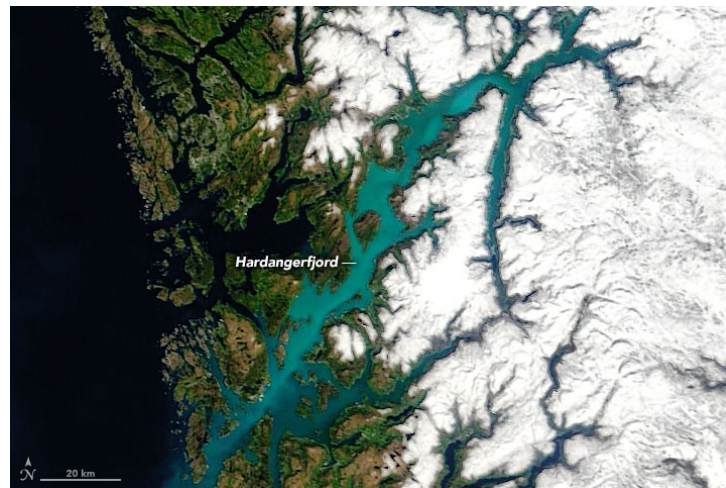


Figure 2.8: Satellite image of an algal bloom in Norway in 2020 captured by NASA's Terra satellite [21].

2.3.2 Remote Sensing Detection of HABs

In order to minimize the impact of HABs, timely detection is critical. Satellite remote sensing monitoring can assist in this and allow for regular observations that could be used to determine if an algal bloom is beginning to develop at a particular location. Real-time or advance notice of HABs allows for proactive measures and the implementation of mitigation strategies to protect aquaculture assets, water supplies, and marine ecosystems.

A commonality between species of algae is the presence of chlorophyll in the organisms. Chlorophyll produces a distinct spectral signature, with the absorption spectrum marked by two prominent peaks within the visible spectrum, usually around the wavelengths 450 nm and 650 nm. This is important for remote sensing detection of HABs because it allows phytoplankton that cause HABs to be broadly identified by two distinct spectral peaks characteristic of chlorophyll. Figure 2.9 illustrates two types of similar absorption spectrum patterns produced by chlorophyll.

Subtle variations in the spectra and peak positions of chlorophyll spectra also have the potential of providing insights into the specific phytoplankton species present in an algal bloom. However, detecting HABs presents challenges due to variations in absorption spectrum of algae, which can be influenced by environmental factors, weather, illumination conditions, growth conditions, and optical properties. It is important to note that the presence of algae does not necessarily indicate the occurrence of a HAB event, as the toxicity of algae can be dependent on specific growth conditions and species of algae [22]. These problems delve into remote sensing problems and biology that are outside the scope of this report. Using knowledge in these areas, however, it is possible to develop optical algorithms capable of measuring chlorophyll in terms of concentration based on satellite retrieval values [23] [24] [25].

Monitoring and tracking algae using optical remote sensing instruments from space is already well-established. Algae concentration estimates are already regularly produced by the European Space Agency's multispectral Sentinel-3 Earth observation satellite [25]. Another example of algae monitoring is carried out by the Geostationary Ocean Color Imager (GOCI), a Korean geostationary satellite dedicated to ocean monitoring. Unlike the hyperspectral imaging capabilities of HYPSON-1, GOCI is a multispectral imager. The primary advantage of GOCI lies in it being a geostationary satellite, as it remains fixed over a single point on the Earth's surface (in this case, the Korean Peninsula).

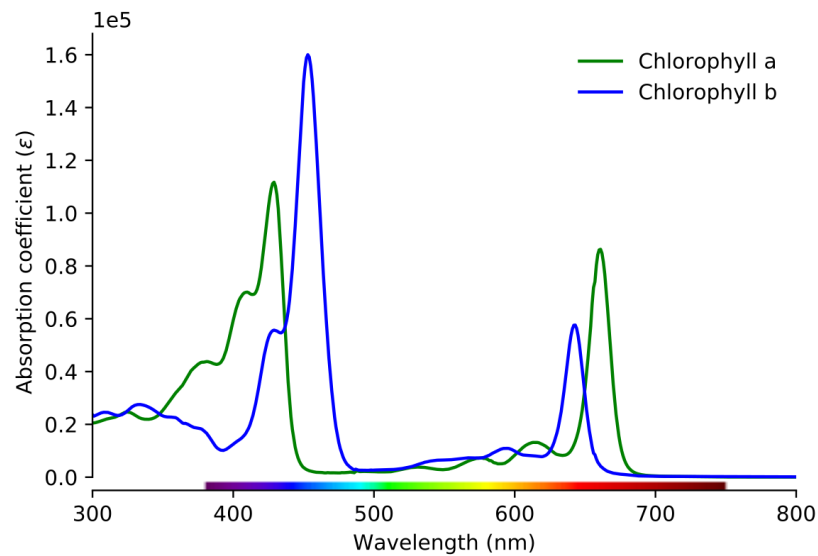


Figure 2.9: Examples of chlorophyll-a and chlorophyll-b spectra [26].

This GOCI to provide frequent observations with a one hour temporal resolution. [27]

In one study, by Huang et al. (2015) [28], observational data from GOCI was used to estimate and track chlorophyll-a concentrations in Lake Taihu in China over periods of several days. Using captures taken over a span of three days, the multispectral GOCI instrument was used to calculate and map chlorophyll-a concentrations within the lake. Algae concentrations were manually mapped and compared in each image. The study demonstrated that satellite imagers could be used to track the spatial and temporal evolution of algae in a body of water but did not directly apply change detection techniques in the analysis.

One concept in the remote sensing detection of HABs, is the idea of an observational pyramid. In an observational pyramid, HAB monitoring satellites would play a role in a much larger monitoring system. HYPSON-1 or another similar satellite could be integrated with other ocean monitoring assets such as aerial drones, autonomous vessels, bouys, or sensors placed at specific areas of interest. By sharing and exchanging data, these assets could achieve a more advanced monitoring of the ocean and HABs. [29] [1]

2.4 Anomaly Detection

Anomaly detection is a process that aims to distinguish unusual patterns and observations in a set of data. Anomalies are defined as patterns that deviate from the normal expected behavior of the data. The nature of anomalies depends on the context of the problem but in general a feature in a set of data is considered an anomaly if it is important for purposes of data analysis. This makes anomaly detection distinct from other techniques such as noise removal which aims to remove extraneous data during analysis [30]. For remote sensing applications anomaly detection is useful to revealing and identifying certain types of natural and artificial phenomena present in remote sensing imagery. Popular methods of performing anomaly detection for remote sensing data include RX detectors, Gaussian mixture models (GMM), and Principal Component Analysis (PCA). [31]

Anomaly detection is an important tool for detecting and monitoring the occurrence of HABs in bodies of water. Because HABs are a relatively sudden and excessive accumulations of toxic algae in a lake or the ocean, they can be treated as an anomaly compared to the normal environment. Hyperspectral imaging provides a means to detect HABs, since they exhibit distinct spectral signatures that differ from the background ocean spectra. Several challenges exist in anomaly detection, however, including determining the number of anomalies present and interpreting or labeling the detected anomalies. A major challenge in anomaly detection is the limited availability of hyperspectral observations of HABs. HABs are relatively rare and short lived events which has limited the amount of satellite observations of the events. This limitation rules out the use of deep learning methods, which require large amounts of labeled training data. Since there is a lack of training data and no prior knowledge of the anomalies, unsupervised anomaly detection methods that can function without labeled training data are necessary.

Previous work has demonstrated that Independent Component Analysis (ICA) is a suitable algorithm for detecting spectral anomalies in hyperspectral data [3]. ICA can identify, extract, and provide relative spectral information about anomalies. By exploiting a statistical property called non-Gaussianity, ICA can separate anomalous signals from the overall hyperspectral data and enable a search for unusual spectra relative to the background spectrum. This is particularly useful in the context of HABs because as there are various types of algae, pigments, and environmental conditions that can influence the anomalous optical spectra.

ICA not only allows for the localization of anomalies but also anomalies to be characterized by their relative spectral signatures. This distinguishes ICA from normal sub-pixel anomaly detectors like Reed and Xioli (RX) detectors, which cannot retrieve both spectral and spatial information about detected anomalous signals [32]. One notable capability of ICA is its ability to detect anomalies in hyperspectral data without requiring any prior information about the data. This means that there is no need to know in advance how many or what types of anomalies exist in the data. Consequently, ICA eliminates the necessity for training or using training data, allowing ICA to be used more flexibly.

By using detected anomalies and spectral information obtained from ICA, it becomes possible to locate potential HABs in hyperspectral images, particularly with anomalies with a spectral pattern resembling chlorophyll spectral signatures. ICA can also identify other features appearing as anomalies, such as sediments in the water. These abilities of ICA to detect anomalous signals in hyperspectral data, function unsupervised, and provide both spatial and spectral information about detected anomalies were the key motivating factors for choosing ICA for use in change detection applications. The spectral information provided by ICA about anomalies is especially advantageous in change detection, as it allows for multiclass detection opportunities.

2.5 Independent Component Analysis (ICA)

2.5.1 Overview

Hyperspectral imaging allows images to be observed at numerous contiguous spectral bands across the visible electromagnetic spectrum at a high spectral resolution. The high spectral resolution offers a more detailed characterization and analysis of the reflected or emitted light from the imaging targets. Simply put, in hyperspectral imaging, there are more individual spectral bands that carry

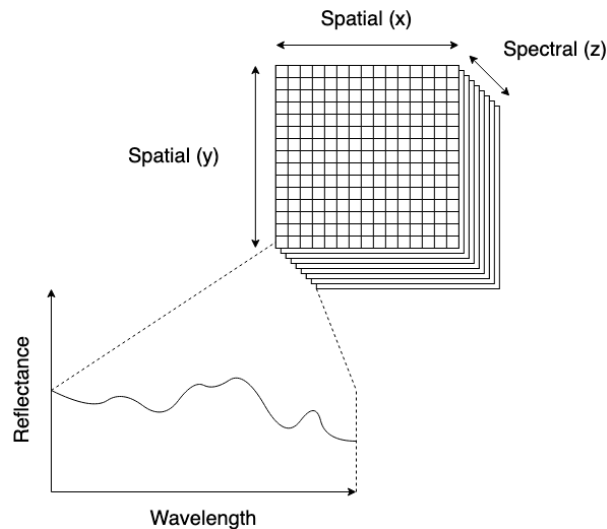


Figure 2.10: Spectrum from a hyperspectral datacube.

information than there are in RGB or multispectral imaging.

In a hyperspectral image, each pixel contains a spectrum representing the reflectance or radiance of light at different wavelengths, as illustrated in Figure 2.10. The interaction of physical materials with visible light varies based on their properties, composition, and chemistry. As a result, different materials exhibit unique spectral signatures due to their distinct interactions with light. Through analysis, these spectral signatures allow for the identification and characterization of various materials captured by multispectral and hyperspectral instruments. In order to properly identify materials and extract information from hyperspectral data, understanding spectral signatures is crucial. By comparing observed spectra to the spectra of known materials, either through a spectral library or by analyzing distinct features like peaks or troughs, materials present in hyperspectral images can be characterized. These spectral comparisons enable the detection of specific materials based on their spectral patterns.

Spectral identification is conceptually simple when there is a single material present. However, when there are two or more materials present, each producing a separate spectral signature, a phenomenon called spectral mixing can occur. Spectral mixing results when multiple spectral signatures from materials located at the same pixel or neighboring pixels within a hyperspectral image are perceived as a combined signal, shown in Figure 2.11. In other words, the observed spectrum at a particular pixel in a hyperspectral image may be influenced by the presence of multiple materials within that pixel or its surrounding area. Spectral mixing is heavily influenced by other factors such as the spatial resolution of the hyperspectral sensor, the physical size of the target, and atmospheric conditions.

Spectral mixing undermines the direct characterization of materials based on observed spectra. It leads to complexity in the interpretation of the hyperspectral data, because the observed spectrum does not directly correspond to a single material but instead a combination of multiple materials. In order to identify individual materials, the effect of spectral mixing must be reversed through a process called spectral un-mixing. Spectral un-mixing decomposes the mixed signal back into separate spectral sources or components that can once again be independently characterized. The spectral un-mixing process enables the identification of multiple materials within a pixel with a mixed spectral signature. The process is shown in Figure 2.12.

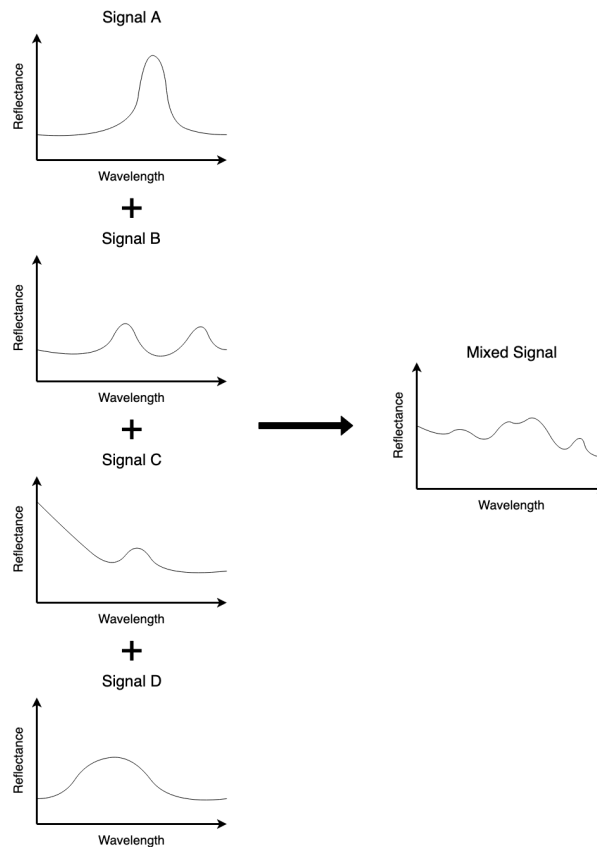


Figure 2.11: Spectral mixing of four different sources, A, B, C, and D.

Fundamentally, spectral mixing and spectral un-mixing are signal processing problems. Algorithms for signal un-mixing and signal decomposition specifically exist for solving similar classes of problems. These algorithms aim to separate a mixed signal into constituent components and determine the contribution or abundance of each component by treating the mixed signal as a linear combination of multiple sources. In the context of hyperspectral imaging, the mixed signal corresponds to the mixed spectrum and the components each correspond to a spectral signature from an individual material or feature within the image.

One of the challenges of spectral un-mixing is the lack of *a priori* information. In the case of mixed data, such as hyperspectral images, information regarding the spectral components present, the specific number of spectral components, and their contributions to the mixed signal are all unknown. It is not possible to anticipate the exact spectral components that will be present in the mixed data. This is sometimes called a blind source separation problem. The lack of *a priori* information in hyperspectral data demands that spectral un-mixing techniques can infer the constituent spectral components by only using the observed mixed data. The inability to anticipate what types of spectral signatures or features are present in data drives spectral un-mixing techniques to use unsupervised and data-driven approaches that can more easily adapt to previously unseen types of data.

As discussed in Section 2.4, a signal un-mixing technique that fulfills the criteria for unsupervised operation and functioning without *a priori* information is Independent Component Analysis (ICA). ICA is capable of both separating mixed signals into their constituent components as well as determining the contribution of each component. These properties make ICA suitable for solving blind source separation problems such as analysis of mixed spectral signals in hyperspectral images. ICA

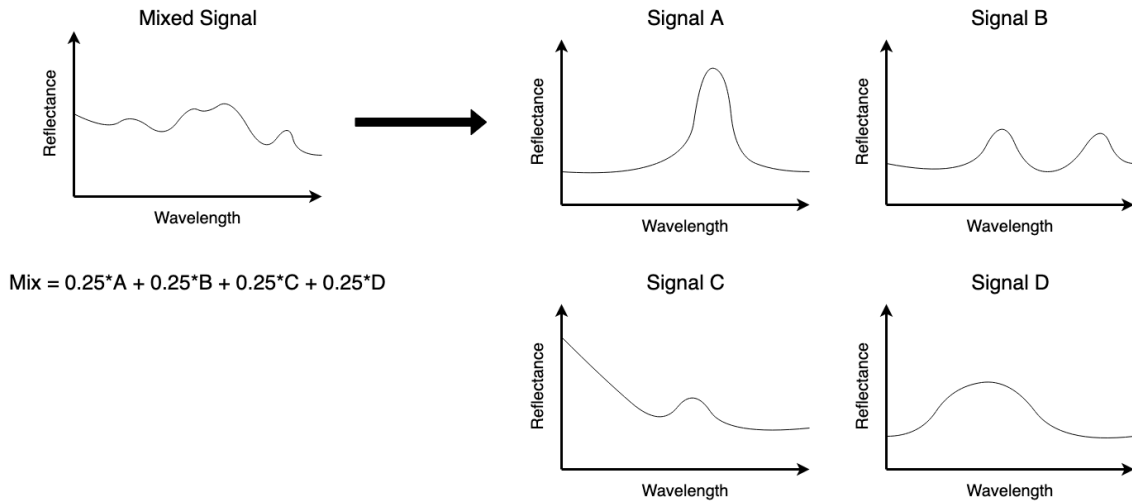


Figure 2.12: Spectral un-mixing of three signals, A, B, C, and D. Together, the signals form a linear combination mixture signal, shown on the left. Each signal is weighted by a value of 0.25 in the mixture signal.

is based on an important principle known as the Central Limit Theorem. The Central Limit Theorem states that the sum of random variables will be closer to a Gaussian distribution than each of the random variables considered independently. ICA exploits the behavior of independent random variables when they are combined with other variables. As more and more variables are combined into the mixture, the collection begins to resemble a normal or Gaussian distribution. With the Central Limit Theorem in mind, the source component signals in the mixture are assumed to be non-Gaussian while noise in the mixture is assumed to be Gaussian. Under this assumption, the components are attempted to be recovered and extracted by identifying non-Gaussian patterns in the mixture signal [33]. There are different implementations of the ICA, however, in this work an algorithm called FastICA is used (Section 2.5.6).

2.5.2 Formulation

ICA treats hyperspectral data as a mixture signal \mathbf{x} . This mixture data consists of a linear combination of source signals s_i that each have a separate weight coefficient \mathbf{a}_i , where index i is the index of the component source signal. In hyperspectral data, each mixture signal sample in \mathbf{x} is represented by a vector x with a length equal to the number of spectral bands or channels. In the case of HYPSON-1, this quantity is 120 bands.

$$\mathbf{x} = \sum_i \mathbf{a}_i s_i \quad (2.1)$$

To simplify notation, the linear combination equation can be expressed as vectors and matrices as is done in Equation 2.2. Using this notation, vectors are written in bold, lower case letters, while matrices are written in bold, upper case letters.

$$\mathbf{x} = \mathbf{A}\mathbf{s} \quad (2.2)$$

Once again, the mixture signal is represented by \mathbf{x} . In the new notation, the vector \mathbf{s} is a vector of source signals and \mathbf{A} is a matrix of the weight coefficients. The weight coefficients in matrix \mathbf{A} , called

the mixing matrix, are used to transform the source signals \mathbf{s} to the linear combination mixture signal \mathbf{x} . The matrix \mathbf{A} is an m by n dimensional matrix with m equal to the number of spectral bands and n equal to the number of source signals or components.

To do the reverse of Equation 2.2 and transform the mixed signal \mathbf{x} to the source signals \mathbf{s} , the equation can be inverted:

$$\mathbf{s} = \mathbf{A}^{-1}\mathbf{x} = \mathbf{W}\mathbf{x} \quad (2.3)$$

To further simplify notation, the inverted matrix \mathbf{A}^{-1} is replaced with \mathbf{W} , a matrix known as the un-mixing matrix. An important thing to note is that \mathbf{A} is not required to be a square matrix. In cases where \mathbf{A} is not a square matrix, then \mathbf{W} is just the generalized left inverse of \mathbf{A} .

At its most basic form, the expression in Equation 2.3 is the mathematical problem that ICA attempts to solve. It takes a mixed signal \mathbf{x} and attempts to decompose it into source signals (or components) \mathbf{s} using \mathbf{W} . ICA approaches this task by attempting to estimate \mathbf{W} .

By obtaining \mathbf{W} , it can be used to recover estimated components, labelled y_i , from the mixture signal \mathbf{x} using Equation 2.4:

$$y_i = \mathbf{w}_i^\top \mathbf{x} = \mathbf{w}_i^\top (\mathbf{A}\mathbf{s}) = (\mathbf{w}_i^\top \mathbf{A})\mathbf{s} \quad (2.4)$$

In this equation, the vector \mathbf{w}_i is the i th row of \mathbf{W} . Because \mathbf{W} is estimated by ICA, the recovered components y_i are not exactly the same as the original source signals \mathbf{s} . Instead, the recovered components y_i are an estimation of the individual source signals by ICA. These components are called the independent components (ICs). The corresponding weights or spectra of the independent components can be recovered from the mixing and un-mixing matrices \mathbf{A} and \mathbf{W} after extracting the components.

In the process of estimating the un-mixing matrix \mathbf{W} , two important assumptions are used [34][33]:

1. **Independence:** The source signals \mathbf{s} are assumed to be completely statistically independent from one another.
2. **Non-Gaussianity:** The individual source signals \mathbf{s} are less Gaussian than the overall mixture signal \mathbf{x} (postulated by the Central Limit Theorem). Put plainly, source or component signals are assumed to be non-Gaussian whereas noise is assumed to be Gaussian.

Using these two assumptions, Independent Component Analysis (ICA) estimates the mixing matrix \mathbf{W} to transform the mixed signal into independent components \mathbf{s} that exhibit maximum non-Gaussianity. Non-Gaussianity is a statistical property referring to the deviation of a dataset distribution from a Gaussian distribution. In order to assess and quantify the level of non-Gaussianity in the independent components, a quantifiable measure is needed in the form of an objective function. A commonly used measure is negentropy, denoted as $J(\mathbf{y})$ (Equation 2.5), which is based on the continuous form of Shannon entropy $H(\mathbf{y})$ (Equation 2.6):

$$J(\mathbf{y}) = H(\mathbf{y}_{\text{gaussian}}) - H(\mathbf{y}) \quad (2.5)$$

$$H(\mathbf{y}) = - \int p(\mathbf{y}) \log_b p(\mathbf{y}) d\mathbf{y} \quad (2.6)$$

In Equation 2.5 and Equation 2.6, \mathbf{y} is a random variable and $\mathbf{y}_{\text{gaussian}}$ is a Gaussian random variable with an identical covariance matrix as \mathbf{y} .

Negentropy quantifies the deviation of a given random variable from a Gaussian distribution. By maximizing negentropy, ICA can identify the components that are the most statistically independent and non-Gaussian. It functions by using the inherent property that a Gaussian random variable will have higher entropy $H(\mathbf{y})$ than a non-Gaussian random variable. The difference in entropy values between a Gaussian and non-Gaussian random variable can be used to construct the objective function for ICA in Equation 2.5. The values produced by negentropy always range between zero (if \mathbf{y} is purely Gaussian) to a positive value (if \mathbf{y} is non-Gaussian), enabling the function to be maximized in an iterative process. In this manner, the objective function can be used by ICA to find independent components.

One of the disadvantages of negentropy is that is not straightforward to compute since it requires pre-existing knowledge of the probability distribution functions of the data. Often, ICA implementations replace negentropy with an approximation or an alternative objective function entirely. This is the approach taken to the objective function in FastICA (Section 2.5.6), which opts for an approximation of negentropy. [34][35]

2.5.3 Preprocessing

ICA has two main preprocessing requirements. These ensure that ICA can accurately and properly extract independent components from the mixture data. [34]

1. **Centering:** The ICA mixed input data \mathbf{x} is required to have a sample mean equal to zero. The process of subtracting the sample mean from the data such that it will have a mean of zero is called centering. Centering ensures that the resulting independent components \mathbf{s} will all have zero mean.
2. **Whitening:** The ICA mixed input data \mathbf{x} must be modified such that the elements within the data are uncorrelated from one another. This step is known as data whitening and is performed after data centering. Whitening helps meet the requirement of uncorrelated data by transforming the data so that the variances of the elements are all equal to one. Following this, the covariance matrix of \mathbf{x} becomes equal to the identity matrix, shown in Equation 2.7:

$$\text{Cov}(\mathbf{x}) = \text{E}[\mathbf{xx}^T] = \mathbf{I} \quad (2.7)$$

2.5.4 Initialization

In some implementations of ICA, the un-mixing matrix \mathbf{W} can be initialized using various methods. A common approach is to use a random initialization, where the matrix elements are set to random values. The choice of initialization method can influence the convergence and performance of the ICA algorithm. The initial un-mixing matrix effectively sets the starting point of ICA and where it begins to search for independent components the data.

2.5.5 Limitations

ICA has two notable limitations. Firstly, ICA is unable to provide information regarding the magnitude or sign (+1 or -1) of the components \mathbf{s} . This ambiguity arises from the unknown nature of both the components \mathbf{s} and the scalar coefficients within the mixing matrix \mathbf{A} . Dividing a component \mathbf{s} by

a scalar value would correspondingly multiply the weight of the component by the same value in **A**. The result of this limitation is that some identifiable independent components have their signs reversed. Fortunately, the effect of this ambiguity is easy to remove by simply changing the sign of the component. Another constraint of ICA is its inability to impose a specific ordering of the components. This is due to the linear combination of weighted sources described in Equation 2.1. Because the terms in the linear summation are commutative, there is no specific order among the independent components (ICs) [34]. Both of these limitations arise directly from the assumptions and formulation of ICA and reflect the nature of working with an unsupervised signal un-mixing technique.

2.5.6 FastICA

FastICA is a commonly used algorithm implementing ICA that was developed by A. Hyvärinen and E. Oja at the Helsinki University of Technology [34]. FastICA is based on a fixed point iteration method for estimating **W** and approximates negentropy rather than measuring non-Gaussianity using the formula in Equation 2.5. The approximation replaces the expression in Equation 2.5 as a more efficient means of computing negentropy. In its simplified form, the negentropy approximation is:

$$J(\mathbf{y}) \propto (E\{G(\mathbf{y})\} - E\{G(\mathbf{y}_{\text{gaussian}})\})^2 \quad (2.8)$$

The function $G(\mathbf{y})$ is one of two non-quadratic functions proposed by Hyvärinen and Oja [34] in Equation 2.9 or Equation 2.10:

$$G_1(\mathbf{y}) = \frac{1}{a} \log \cosh(a\mathbf{y}) \quad (2.9)$$

$$G_2(\mathbf{y}) = -e^{-\frac{\mathbf{y}^2}{2}} \quad (2.10)$$

The coefficient a in Equation 2.9 is an arbitrary coefficient that can be selected in order to modify the objective function. In FastICA implementation its value defaults to 1.

The function in Equation 2.8 can serve as an objective function for ICA. Much like the exact form of negentropy, the approximation is equal to zero for Gaussian random variables and is a positive non-zero value for non-Gaussian random variables. As a result, it can be maximized and used to find maximally non-Gaussian independent components using FastICA's fixed point iterative method.

The non-quadratic functions can also be replaced entirely allowing for the FastICA algorithm to be modified or optimized [34]. By default, the Python implementation of FastICA from SciKit-Learn [36] uses Equation 2.9 as part of its approximation of negentropy. For the rest of this work it can be assumed that FastICA is run with this approximation function.

2.6 Dimensionality Reduction

2.6.1 Overview

As discussed in the limitations of ICA (Section 2.5.5), ICA does not automatically generate independent components (ICs) in a fixed order. This has two effects on the results. Firstly, ICA does not order independent components in terms of their contribution or abundance in the mixed data. Unsurprisingly, the lack of an inherent order in the components leads to computational resources and time spent on extracting independent components contribute little to the mixture instead of more signifi-

cant independent components. Secondly, the absence of a fixed order means that by default ICA will produce a set of independent components in a different order each time the algorithm is run. This is heavily influenced by the values that the un-mixing matrix \mathbf{W} is initialized (i.e. the algorithm's initial conditions). In a sense, these values dictate the starting point where ICA begins searching for independent components within the mixed data.

Having the ability to detect components by significance helps to ensure that most informative independent components are prioritized. This can be used to highlight potentially relevant features in the mixed data. The ordered generation of components is also useful for locating the same spectral signatures or features in two distinct hyperspectral images, such as multiple hyperspectral images in a time series. It allows the same sets of spectral features to be tracked across multiple images in applications like change detection. Finally, fixed order components also enables the repeatability of running ICA on sets data, which in turn can be used to develop and evaluate performance and reliability aspects of ICA-based anomaly detection.

2.6.2 ICA-DR

A combination of two techniques called Principal Component Analysis (PCA) and ICA dimensionality reduction (ICA-DR) was used to enhance ICA and enable the algorithm to produce independent components in a fixed order. ICA-DR is based on the ICA-DR3 algorithm developed in [37] and has been adapted to successfully process hyperspectral imagery [3]. At a high level, ICA-DR3 operates by selecting and initializing the values of the un-mixing matrix \mathbf{W} so that the ICs are generated in order of their significance. This is achieved in hyperspectral data by leveraging an algorithm called Automatic Target Generation Procedure (ATGP) [37] [38]. ATGP is a procedure for automatic target recognition in hyperspectral images.

The process of selecting the un-mixing matrix values consists of several steps. First, a dimensionality reduction technique called Principal Component Analysis (PCA) is run on the hyperspectral data. PCA reduces the dimensionality of the hyperspectral data and transforms it in to a smaller data representation consisting of principal components. The principal components generated by PCA are ordered in their significance to the dataset based on variance. Next, the principal components are used as input to the ATGP algorithm. The ATGP attempts to construct a initial un-mixing matrix that incorporates information about the principal component features produced by PCA and their order of significance. The un-mixing matrix is then used to initialize ICA to process the original hyperspectral data. Since PCA consistently generates identical principal components when given the same set of data, ATGP will similarly yield the same initial un-mixing matrix when provided with the same set of principal components. This enables the ICA initial conditions to be consistently set each time and to extract the same independent components in a fixed order of significance. The PCA and ATGP algorithms are discussed in detail in Section 2.6.3 and Section 2.6.4.

2.6.3 Principal Component Analysis (PCA)

PCA serves as a dimensionality reduction technique by linearly transforming the data into a new coordinate system. In the PCA transformed space, the data is represented with a reduced number of newly created and uncorrelated variables. The objective of PCA is to find the new variables as linear combinations of the original variables that maximize variance [39].

PCA begins by describing each sample \mathbf{x} consisting of p variables as a linear combination. The values a_j represent weights in the linear combination. The sum of the samples and their weights can be rewritten as matrices:

$$\sum_{j=1}^p a_j \mathbf{x}_j = \mathbf{X}\mathbf{a} \quad (2.11)$$

The variance of the linear combination can be calculated and derived into a formula incorporating the covariance matrix \mathbf{S} :

$$\text{Var}(\mathbf{X}\mathbf{a}) = \|\mathbf{X}\mathbf{a}\|^2 = \mathbf{a}^T \mathbf{X}^T \mathbf{X} \mathbf{a} = \mathbf{a}^T \mathbf{S} \mathbf{a} \quad (2.12)$$

PCA optimizes to maximize variance so the problem is written as a Lagrange multiplier where $\mathbf{a}^T \mathbf{a} = 1$:

$$\mathbf{a}^T \mathbf{S} \mathbf{a} - \lambda (\mathbf{a}^T \mathbf{a} - 1) \quad (2.13)$$

Taking the derivative of 2.13 and the equation equal to zero, the expression becomes:

$$\mathbf{S}\mathbf{a} - \lambda \mathbf{a} = 0 \quad (2.14)$$

Note that Equation 2.14 is an eigenequation can be solved for the eigenvalue λ . From this, the largest eigenvalue λ_k can be used to determine a linear combination that maximizes variance:

$$\text{Var}(\mathbf{X}\mathbf{a}_k) = \mathbf{a}_k^T (\mathbf{S}\mathbf{a}_k) = \mathbf{a}_k^T (\lambda_k \mathbf{a}_k) = \lambda_k \mathbf{a}_k^T \mathbf{a}_k \quad (2.15)$$

The linear combination $\mathbf{X}\mathbf{a}_k$ is denoted as the k th principal component, which corresponds to the k th eigenvalue. If S is a square covariance matrix and all vectors \mathbf{a}_k are orthogonal, then there can be up to p eigenvalues and principal components. For processing hyperspectral images with ICA and ATGP, the principal components generated using this process are used as input for the ATGP algorithm.

2.6.4 Automatic Target Generation Process (ATGP)

Automatic Target Generation Process (ATGP) implements an automatic procedure for identifying spectral targets in hyperspectral images [38]. It was developed alongside the ICA-DR3 algorithm, a method that iteratively generates the initial un-mixing matrix \mathbf{W} used by ICA. The ATGP starts with a set of initial projection vectors. The initial projection vectors are used as input for ATGP and are generated and selected from the PCA-transformed hyperspectral data.

The first projection vector \mathbf{t}_0 is selected by searching for the vector \mathbf{r} with the maximum orthogonal projection:

$$\mathbf{t}_0 = \arg\{\max_{\mathbf{r}}(\mathbf{r}^T \mathbf{r})\} \quad (2.16)$$

The resulting vector \mathbf{t}_0 is added to a matrix \mathbf{U} of undesired spectral signatures. This has the eventual effect of hiding the undesired spectral signatures from the un-mixing matrix and ICA. The matrix \mathbf{U} is used to transform the remaining data and remove the undesired spectral signatures from the data. This removal is performed by computing an undesired spectral signature annihilator using \mathbf{U} and the identity matrix \mathbf{I} :

$$P_{\mathbf{U}}^{\perp} = \mathbf{I} - \mathbf{U}(\mathbf{U}^T \mathbf{U})^{-1} \mathbf{U}^T = \mathbf{I} - \mathbf{U}\mathbf{U}^+ \quad (2.17)$$

With the annihilator expression calculated, it can be applied to the data and used to remove spectral

signatures that have already been detected by ATGP and the maximum orthogonal projection search. The removal of the detected spectral signatures allows the process to continue searching for other significant spectral signatures present in the data. Equation 2.16 is used again, this time with the annihilator from Equation 2.17 applied to the data:

$$\mathbf{t}_i = \arg\{\max_{\mathbf{r}} (P_{\mathbf{U}_i}^\perp \mathbf{r})^T (P_{\mathbf{U}_i}^\perp \mathbf{r})\} \quad (2.18)$$

As more and more projection vectors are found, the matrix \mathbf{U} is continuously expanded, as seen in Equation 2.19.

$$\mathbf{U}_i = [\mathbf{t}_0, \mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_i] \quad (2.19)$$

Alongside this, $P_{\mathbf{U}_i}^\perp$ is also updated using \mathbf{U}_i and Equation 2.17. The entire iterative process continues until it yields n spectral targets, where n corresponds to the number of components selected for the PCA transformation. The final output matrix, denoted as \mathbf{U}_n , consists of n targets represented by $[\mathbf{t}_0, \mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_n]$. Finally, the matrix \mathbf{U}_n is returned and used as the initialization matrix \mathbf{W} for ICA [38].

2.7 Change Detection

2.7.1 Overview

Change detection is the process of comparing data from different times drawn from a time series dataset. The purpose and aim of change detection is to detect and analyze differences in data over time. For remote sensing applications, the data comes in the form of aerial or satellite imagery. Change detection can be applied to hyperspectral and multispectral images as well as RGB or monochrome images. Change detection techniques are applied to sets of images called time series datasets which are composed of images acquired at different points of time. Using change detection approaches, the images drawn from time series datasets are analyzed to identify and map pixels, objects, other features that have undergone a change, revealing information about natural and human-made phenomena that have occurred within the region of interest. Typically this application of change detection is used to identify patterns of change in land use, agriculture, forestry, and urban planning. However, HYPSON-1 is an ocean color observing satellite. In this case, HYPSON-1 captures are used to monitor for changes such as algae growth seen in bodies of water such as oceans, lakes, and fjords. [4]

Hyperspectral change detection is an area of significant interest, because of the higher spectral resolution in hyperspectral imagery. This form of imaging provides more extensive spectral information about targets than conventional multispectral and RGB imaging. As a result, hyperspectral change detection has the potential to detect and observe much more subtle changes with enhanced detail. Historically, there have been more multispectral and RGB datasets so research and development have focused on change detection using data from those remote sensing techniques rather than hyperspectral imaging. Recently, however, new hyperspectral imaging satellites including HYPSON-1 have entered into operations and are expanding the amount of hyperspectral data. [4] [8]

2.7.2 Challenges

For hyperspectral change detection, there are several unique challenges. One of the most significant is the high dimensionality of the data, due to the many spectral bands present in hyperspectral data.

The problem of efficiently generating, manipulating, and processing high dimensional hyperspectral data is magnified when a sequence of images is involved, such as in a time series dataset. Hyperspectral datacubes require large amounts of storage space compared to conventional RGB or multispectral images and are also slow to distribute due to the large file size.

Another difficulty is high redundancy of the data. In hyperspectral data with a high spectral resolution, neighboring spectral bands may share much of the same information and as a result introduce unneeded redundancy to the data. This further contributes to the problem of high dimensionality without contributing additional useful spectral information. The large raw amount of data makes identifying changes difficult and complex. For example, HYPSON has 120 spectral bands and a 956 pixel by 684 pixel spatial resolution. Depending on the detection problem, some of the spectral bands may be unnecessary and as a result contribute to processing inefficiencies.

The large amount of data contained in hyperspectral images becomes even more significant when the data spans across multiple captures as multitemporal dataset. In some applications the problem of the large amount of data and spectral bands can be solved by simply dropping spectral bands and keeping only a subset of useful wavelengths. The difference comparison-based spectral change vector (SCV) method is one possible approach to the high dimensionality issues of hyperspectral data. Multitemporal data is used to construct change vectors thereby reducing the total amount of data handled by the change detection algorithm. [4]

Other problems associated with hyperspectral data include low signal-to-noise ratio (SNR) and atmospheric effects. These problems arise from the noise and artifacts caused by the hyperspectral sensor itself as well as the influence of weather and atmospheric conditions such as water vapor and cloud cover. Consequently, these factors can complicate the task of detecting and distinguishing the desired spectral patterns from the rest of the hyperspectral data. [8]

While not unique to hyperspectral imaging specifically, spectral variability is another challenge present. This poses a problem if a specific change is targeted for detection and observation. Various environmental factors and materials can alter absorption and reflectance spectra picked up by imaging sensors. For example, during the detection of phytoplankton from satellite-based multispectral sensors, dissolved sediments, organic materials, as well as simple environmental conditions can significantly influence the ocean-color and spectral shape of spectral signals indicating phytoplankton and algal blooms. [22]

2.7.3 Binary and Multiclass Change Detection

Change detection can be categorized into two main categories: binary change detection and multiclass change detection. Binary change detection is the simplest change detection category and attempts to identify any kind of change, regardless of its source or location. Binary change detection effectively classifies the change in every pixel as either true (a change is detected) or false (a change is not detected). In some literature, changed pixels are sometimes referred to as the "foreground" and unchanged pixels are referred to as the "background".

In contrast, the other category of change detection is multiclass change detection. It extends binary change detection to use more than two classes. Instead of deciding simply if a change occurred and labelling it as true or false, multiclass change detection attempts to separate different types of change into three or more classes which can correspond to different causes of change. Overall, multiclass change detection is an extension of binary class change detection where binary change de-

tection is a special case of multiclass change detection with only two classes. In multiclass change detection, it is difficult to have a complete multiclass temporal ground reference dataset. As a consequence, multiclass change detection methods that require supervised training are ruled out, leaving unsupervised or semi-supervised methods as the preferable option. [4]

2.7.4 Supervised and Unsupervised Change Detection

Supervised and unsupervised change detection techniques are distinguished from each other by the need for training data in order to perform change detection analysis. Supervised change detection refers to change detection algorithms that rely on label training data. Training data can be important for multiclass change detection techniques since it can provide accurate information about the nature of an pixel location or object in terms of the material, composition, and spectral signature. One of the problems with supervised techniques is the difficulty of assembling a ground truth dataset. As is the case for hyperspectral images of HAB, it can be difficult to construct a training dataset of images featuring a specific type of feature that is under investigation. This is magnified by the limited number of hyperspectral image datasets and the spatial and temporal localization of the images that do exist. [4]

In contrast to supervised change detection, unsupervised change detection does not require training data. Unsupervised change detection techniques can directly perform change detection analysis without needing to be trained on labelled training data. Depending on the technique used, it can be difficult to use unsupervised change detection for multiclass problems since the number of distinct types of changes in a dataset is unknown at the beginning. Consequently, not only do unsupervised methods need to identify the location and amount of change, they also need to determine the number of types of change and distinguish between them without intervention [4]. The topic of this paper, ICA-based change detection methods, are unsupervised techniques meaning that they do not require training data or labelled data to train the ICA algorithm.

The degree of automation in change detection is somewhat related to the supervised and unsupervised distinction. The degree of automation can be classified as fully automatic, semi-automatic, or manual change detection, Automatic change detection is ideal since it requires the least amount of human intervention. As a result, automatic change detection is desirable for automated ocean monitoring systems. [4]

2.7.5 Bitemporal and Multitemporal Change Detection

Change detection approaches can vary based on the span of time considered as well as the number of time series images used in the analysis. This provides the basis for separating change detection into two categories: bitemporal change detection and temporal trajectory analysis or multitemporal change detection [40]. In this report, temporal trajectory analysis is referred to as multi-temporal change detection. This is done to reflect that it performs change detection across a continuous series of images (multitemporal) rather than pairs of images (bitemporal).

Bitemporal change detection performs change detection on pairs of temporally separated images. This is the simplest of the two categories since it is comparing only two images at once. Most previous works deal with bitemporal change detection, only trying to determine if a change as occurred between two images. In comparison, multitemporal change detection performs change detection on

a series of temporal images. Multitemporal change detection aims to identify changes as well as track the progression of changes over time. Multitemporal change detection can be achieved by processing all the images together or by processing the images in pairs using a bitemporal technique and repeating the change detection until all the images have been processed. The latter method for multitemporal change detection is simpler since it simplifies and breaks the change detection process into manageable steps which can leverage bitemporal change detection methods. [4] [40]

Multitemporal change detection attempts to characterize the progression and trend of changes in the data. Naturally, this requires more images than bitemporal change detection. One current challenge for the HYPSON-1 project is to generate time series datasets that consist of more than two or three images while still maintaining good spatial overlap. HYPSON-1 captures cover a narrow swath so multiple captures must align closely to achieve good spatial overlap. As a result of this, bitemporal datasets were utilized with the ICA-based change detection techniques.

2.7.6 Preprocessing

Correct preprocessing is a vital component of hyperspectral change detection. When working with multitemporal datasets, it is essential to process all hyperspectral images uniformly. Processing data for use in change detection has several different angles. Among these, spectral calibration ensures accurate alignment of spectral sensor bands with their respective spectral wavelengths. Radiometric and geometric corrections are equally important. These corrections address sensor artifacts and attempt to remove noisy bands and striping, preventing their interference with the change detection algorithm where they can show up as detectable changes and reduce accuracy. Geocoding, the process of adding geospatial information and context to the images, is also necessary. It ensures proper geographic alignment and georeferencing among the hyperspectral images. Finally, atmospheric correction minimizes the influence of atmospheric effects on the data, further enhancing the accuracy of the change detection process. [4]

2.7.7 Change Detection Categories

Change detection techniques can be categorized in multiple ways based on various criteria and characteristics. In Jianya et al. (2008) [40], the authors categorize change detection methods into seven categories: direct comparison, classification methods, object oriented methods, model methods, time-series analysis, visual analysis, and hybrid methods. In a much simpler categorization, the authors in Zhou et al. (2018) [41] separated change detection methods into two categories, pixel-based and object-based CD methods. Finally, in Lu et al. (2004) [42], a set of seven categories were used: algebraic, transformation, classification, advanced models, geographic information system (GIS), visual analysis, and other methods. These varying methods of categorization highlight that there is no real consensus for categorizing change detection techniques.

In this report, change detection is categorized into three different paradigms: pixel-based change detection, feature-based change detection, and object-based change detection. These paradigms are distinguished by the level at which they analyze the data; for instance, there is a significant distinction between analyzing changing features or objects seen in an image versus analyzing changing the individually changing pixels.

This report also categorizes change detection by approach. The categorizes discussed here are

broadly based around the categories introduced in Jianya et al. (2008) [40], Zhou et al. (2018) [41], and Lu et al. (2004) [42]. These approaches include algebraic, transformation, classification, and hybrid techniques.

Pixel-based Techniques

Pixel-based change detection are techniques that analyze changes at the pixel level in imagery. At each pixel, the spectral values from two or more images are compared to locate and identify changes in the images. Since pixel-based change detection operates at the pixel level and does not consider neighboring pixels, it lacks spatial context that is sometimes useful for detecting changes. A common method of pixel-based change detection is Change Vector Analysis (CVA) in which spectral change vectors (SCVs) are used to quantify the magnitude and direction of change at each pixel location. Another much simpler technique is image difference, where values in the first image are subtracted from the second image, yielding a change map.

Pixel-based change detection usually operates on the raw data and does not attempt to classify or transform parts of the image before attempting to find changes. Techniques such as post-classification comparison (PCC) where pixels from each image are first classified into different classes and then compared are not inherently pixel-based but may incorporate pixel-based comparison strategies such as differencing. [42] [41]

Feature-based Techniques

Feature-based change detection is another class of techniques. Rather than directly analyzing and comparing the pixels in each image individually, feature-based change detection extracts relevant features and characteristics from the images and then performs a comparison of the features. The change map resulting from this procedure is associated only with the specific feature that was used to create it. [4]

In general feature-based change detection operates on a model of transforming images from the image domain to a feature domain. Feature-based techniques include transformation methods such as the K-T transformation, multivariate alteration detection (MAD), principal component analysis (PCA), and independent component analysis (ICA). [8] [43] [42] [44]

Feature-based change detection is a more complex approach compared to pixel-based change detection. However, it provides the capability to distinguish and separate different types of changes. This characteristic makes feature-based change detection well-suited for applications that require multiclass change detection.

Object-based Techniques

Object-based change detection are methods used to analyze changes in specific objects or spatial regions of interest within images. Instead of focusing on individual pixels, object-based change detection operates on groups of pixels which form objects or continuous spatial units. These objects form the underlying units used in the change detection process instead of pixels. Object-based change detection can identify objects through processes such as clustering or image segmentation. By analyzing the spectral properties and spatial relationships of identified objects, changes can be detected. [41]

Object-based change detection offers advantages such as the ability to incorporate spatial contextual information which is lacked in pixel-based change detection. This is important for applications such as land cover analysis where changes at a regional spatial level are of interest. Like feature-based change detection, object-based change detection is suitable for multiclass change detection applications. Different types of changes can be distinguished by analyzing the properties of different objects identified in the images. The objects can be classified into different types of changes, either before or after the image segmentation or clustering process. [45] [41]

Algebraic Techniques

Algebraic change detection techniques refer to the use of mathematical equations and statistical methods to directly compare and analyze pixel values or transformed components with the goal of detecting and characterizing changes between multiple images. Algebraic change detection methods encompass techniques such as image differencing, image regression, image ratioing, and change vector analysis (CVA). Algebraic change detection approaches also include direct comparison methods when applied directly to the original untransformed or unclassified dataset. In some cases, algebraic change detection can be a component of other types of change detection. For example, in change detection using a transformation method, such as principal component analysis (PCA), algebraic methods are used to analyze changes in the resulting components. [42]

Changes are typically identified by comparing the results of the comparison to a predetermined threshold value. Algebraic change detection methods are well-suited for bitemporal change detection applications that involve comparing two images. One limitation of algebraic methods however is the challenge of determining appropriate threshold values that accurately indicate whether a change has occurred. [42] [40]

Transformation Techniques

Transformation techniques offer an alternative to algebraic change detection approaches. Transformation techniques focus on extracting features from images, making them inherently feature-based in nature. Transformation techniques include change detection algorithms incorporating methods such as K-T transformations, multivariate alteration detection (MAD), principal component analysis (PCA), and independent component analysis (ICA) [42] [4]. Algebraic methods may be incorporated and combined with transformation techniques. After extracting features from the data, the resulting features can subsequently be utilized in an algebraic-based post-analysis comparison approach to identify and highlight areas where various types of changes are taking place. Through these steps, a change map indicating changed regions of the images can be produced. [42]

Transformation techniques serve as a means of dimensionality reduction. They operate within the feature domain and on feature components rather than on a pixel level. By taking advantage of transformation techniques, the amount of data needed to capture and represent changes can be significantly reduced. Ultimately, the purpose of transformation methods is to transform high dimensionality data into a compressed set of components.

Although transformation techniques have advantages in dimensionality reduction, there are some drawbacks to the approach. Interpreting the transformed components is a challenge with transformation approaches. Depending on the number of changes and number of components, identifying

and labelling the most significant components is time consuming and may require user interaction. This becomes a major problem for applications where change detection is required to be capable of identifying anomalous changes as well as operate unsupervised and automatically [42] [4]. Another major disadvantage of transformation methods is that transformation-based change detection can not determine the number of multiclass changes. Additionally, depending on the method of post-analysis comparison, a threshold value to identify changes is still required [4]. Despite these problems, transformation methods remain a powerful approach due to their dimensionality reduction abilities.

Transformation techniques include both PCA and ICA [39]. PCA-based transformation change detection for satellite imagery is an already established use case. Fung and LeDrew (1987) [46] utilized PCA to study land-cover changes using Landsat multispectral images [46]. More recently, Ortiz-Rivera et al. (2006) used a variation of PCA, temporal PCA (TPCA), to process pairs of hyperspectral imagery. In their approach, the pairs hyperspectral images were stacked spatially, processed with PCA, and a decision rule applied to detect changes [47]. Another transformation technique uses independent component analysis (ICA). One of the advantages of ICA is that it is an unsupervised technique. ICA does not require training data and can be performed on new captures without having to change or modify the algorithm. Because of the unsupervised change detection techniques are of interest due to their suitability for automatic change detection. [8] [43]

Classification Techniques

Classification techniques encompass change detection techniques that use classification algorithms to identify and categorize changes in features of interest. There are various classification techniques available, such as Post-Classification Comparison (PCC), which is also known as independent image analysis, expectation-maximization (EM) methods, and methods utilizing classification neural networks or support vector machines (SVM). Post-Classification Comparison (PCC) is a commonly used method for change detection, involving the classification of each image independently, followed by a comparison to identify changes. An alternative approach to PCC is direct multivariate classification, which involves stacking the images or datacubes and applying a classification algorithm directly on the combined multitemporal dataset to detect and highlight various classes of change. Direct multivariate classification enables for some spatial context and information to be preserved. [42] [48]

Classification techniques can be categorized as either supervised or unsupervised, depending on the specific classification method. Unsupervised techniques are important for hyperspectral change detection due to the challenge of having a sufficient ground truth dataset to train classification models on. [42] [4]

Hybrid and Other Techniques

Some change detection methods cannot be accurately categorized, either because they combine elements from multiple techniques, or because the approach does not neatly match any of the predefined categories. These techniques exhibit characteristics that make them challenging to classify. Hybrid approaches refer to instances where multiple change detection techniques are combined to create a distinct new technique incorporating elements from the original techniques. This allows for different strengths and characteristics of techniques to be selected and combined in order to create

novel approaches to change detection. [42]

2.7.8 Change Detection Methods

Overview

This section provided a description of several common or relevant change detection methods. Each method is categorized based on the change detection paradigm it uses (pixel, feature, or object-based) and by its technique (algebraic, transformation, classification, hybrid, or other).

Data Differencing

- Techniques: Pixel-based; Algebraic

Differencing is a widely used technique for change detection. It involves subtracting the pixel values from two hyperspectral images acquired at different times to identify areas of significant change. The resulting difference image shows the spectral variations between the two time points, making it easier to detect changes in the scene. Differencing can be combined with other techniques such as PCA and ICA [49] [4]

Change Vector Analysis (CVA)

- Techniques: Pixel-based; Algebraic

CVA involves computing a spectral change vector (SCV) for each pixel in the hyperspectral image. For each pixel, a vector consisting of values equal to the difference of each spectral band between the two images is constructed. The SCVs are used by the change detection algorithm rather than the images themselves. [4]

Image Stacking

- Techniques: Pixel-based or Feature-based; Algebraic, Transformation, or Hybrid

Another change detection technique is image stacking. In image stacking, multitemporal hyperspectral images are combined or stacked in an additional dimension. The stacking process creates a new composite hyperspectral datacube that extends the dimensionality of the data. The dimension in which the images are stacked can be in a spatial, spectral, or even a new temporal dimension. [4]

Post-Classification Comparison (PCC) or Independent Image Analysis

- Techniques: Pixel-based or Object-based; Classification

In independent image analysis, pixels in images are classified based on their spectral signatures. For this step, each image is processed independently from each other. After making the classifications, the differences between the various classes are compared across time bitemporally. This method is also known as Post-Classification Comparison (PCC). PCC can rely on classification truth data to develop a classification algorithm. In some cases this is a disadvantage if classification truth data does not exist for the hyperspectral dataset. In this case, the PCC method must be capable for determining

the number of classes if it is being used for a multiclass change detection problem [4]. Due to the reliance on ground truth data, PCC is not the most common of hyperspectral image change detection. Comparison operator and image stacking approaches are more widely used for hyperspectral image change detection [4].

Principal Component Analysis (PCA)

- Techniques: Feature-based; Transformation

PCA is an unsupervised method of dimensionality reduction which changes the data by linearly transforming it to a new coordinate system. After transforming the data, the newly transformed data can be represented with a reduced set of variables. PCA achieves dimensionality reduction by finding new variables as linear combinations of the untransformed variables while simultaneously maximizing variance [39]. The principal components generated by PCA can be used in change detection algorithms. This can be done by applying PCA to hyperspectral images individually and producing components for comparison or by stacking multiple hyperspectral images and processing them together with PCA. Both methods can support multiclass change detection. [49]

Independent Component Analysis (ICA)

- Techniques: Feature-based; Transformation

ICA is an unsupervised method that transforms mixed data into independent components. It converts data from the image domain to a feature domain. ICA can be used in a similar manner as PCA and be combined with other change detection techniques [43]. One interesting ICA-based approach proposed in S. Liu et al. (2012) [50]. It combines ICA with a hierarchical spectral dimensionality reduction algorithm. ICA is used to extract and detect changes in pairs of images at varying spectral resolutions by merging neighboring spectral bands. Another approach is spatio-temporal ICA [51] which attempts to find temporal and spatial components in hyperspectral data. Despite its potential for change detection, ICA is not as popular or widespread as PCA. ICA is described in detail in Section 2.5.

2.8 Sustainable Development Goals

The 2030 Agenda for Sustainable Development is a framework adopted by the United Nations (UN) in 2015 that seeks to address global social, economic, and environmental challenges. It consists of 17 Sustainable Development Goals (SDGs) that encompass interconnected objectives including eradicating poverty, climate action, and promoting social equality, listed in Figure 2.13. The SDGs are intended to serve as a guideline for global development by countries over the next decade through 2030. [52]



Figure 2.13: The 17 Sustainable Development Goals [52].

Environmental monitoring satellites like HYPSON-1 and others play a significant role in Earth observation, and their application aligns closely with several Sustainable Development Goals (SDGs). The SDGs include SDG 6: Clean Water and Sanitation; SDG 9: Industry, Innovation, and Infrastructure; SDG 13: Climate Action; SDG 14: Life Below Water; and SDG 17: Partnership for the Goals.

2.8.1 SDG 6: Clean Water and Sanitation

The Clean Water and Sanitation development goal aims to ensure access to clean drinking water for all and to protect water-based ecosystems. One area where satellite observation and change detection play important roles is the detection and monitoring of Harmful Algal Blooms (HABs) in oceans and lakes. In fresh water lakes, HABs can pose serious threats to drinking water quality. Improved change detection techniques that can identify and track the progression of HABs near drinking water supplies could be used to help mitigate the effects and ensure access to safe drinking water.

2.8.2 SDG 9: Industry, Innovation, and Infrastructure

The Industry, Innovation, and Infrastructure goal promotes sustainability in industry activities and the development of sustainable infrastructure. Better satellite monitoring capabilities can provide information to help established industries such as fishing, aquaculture, and transportation to transition to more sustainable practices and avoid negative environmental impacts.

2.8.3 SDG 13: Climate Action

The Climate Action development goal seeks to promote changes to prepare and mitigate climate change. Satellite observation and processing algorithms enable more comprehensive monitoring and understanding of the effects of climate change. Additionally, they can help to strengthen surveillance

of natural hazards and natural disasters such as HABs. They can also add to the ability to plan and set policy for responding to the effects of climate change.

2.8.4 SDG 14: Life Below Water

The Life Below Water sustainable development promotes sustainable fishing, reducing pollution, and protecting marine ecosystems. It also strives to increase knowledge of ocean health and biodiversity. Satellite observation and hyperspectral image change detection improves the management and environmental monitoring of marine resources. It can lead to a better understanding of aquatic ecosystems. Furthermore, detection of hazards like HABs can help protect marine food stocks and aquaculture assets such as fisheries.

2.8.5 SDG 17: Partnership for the Goals

The Partnership for the Goals development goal seeks to encourage international cooperation and the sharing of science, technology, and other knowledge. The efforts made in developing hyperspectral change detection algorithms, along with the broader work carried out at NTNU and the Small Satellite Lab, can play a significant role in encouraging international collaboration and facilitating the transfer of knowledge. An essential aspect of achieving this objective involves ensuring open data access and promoting the transparent sharing of work.

Chapter 3

Implementation

3.1 Overview

The implementation of ICA-based change detection consisted of several components. In the first component, HYPSO-1 hyperspectral data was calibrated, processed, and combined using a processing pipeline to form multitemporal time series datasets. The processing pipeline ensured that all data was aligned and resampled to the same spatial resolution, resulting in time-indexed datasets that could be used for bitemporal change detection. A second component of the implementation was composed of development done on FastICA to prepare it to be used with change detection algorithms. Following this, a third implementation component consisted of tasks related to the change detection processing itself, included loading desired pairs of bitemporal imaging, data reshaping and preprocessing, generating ICA components, and applying different change detection methods to the data. The final component dealt with validation data. It involved obtaining suitable data for comparison and analysis with the HYPSO-1 change detection results. The following sections in this chapter provide more detail about each part of the implementation along with a summary about the software environment and HYPSO-1 data.

3.2 Software Environment

The code written to perform the tasks in this project were developed and run using a software environment based around Python version 3.9.13. The software environment included several additional Python libraries which provided additional functionality. This included mathematical function libraries such as NumPy [53] and SciPy [54]. Matrix and data manipulation operations were performed using the Numpy, xarray [55], and Pandas [56] libraries. The masked array module in NumPy was also used [57], primarily for applying land masks to hyperspectral data.

A variety of packages were also utilized to handle geospatial tasks, such as georeferencing, geometric transformation, and resampling. These libraries included GeoPandas [58], Rasterio [59], pyproj [60], skimage [61], and GeoCube [62]. In tasks involving ESRI shapefiles and geospatial data, Shapely [63] and GDAL/Fiona [64] were also used. SatPy [65] was used in a limited capacity to provide an efficient resampling solution with pyresample [66] for the validation data. To implement the ICA algorithm, the SciKit-Learn library was used. It provided the necessary preprocessing functions for data centering and whitening in addition to a well-supported implementation of the FastICA al-

gorithm [36] [67]. Plotting capabilities were handled with the Matplotlib and Basemap libraries used together. The combination of Matplotlib and Basemap allowed for the inclusion of geospatial data, such as latitude and longitude lines, in plots of ICA components and change maps. [68] [69]

The code written for this project was uploaded and hosted on NTNU SmallSat Lab's internal GitHub software repository. Code written for specifically for time series dataset processing and change detection uses was committed to the `hs-time-series` GitHub repository. Within this repository, the code was integrated into a custom Python package named `hsts` which provided a set of functions for generating time series datasets that could be invoked from a Python script or Jupyter notebook. Accompanying the `hs-time-series` project was a separate repository for georeferencing ground control points (GCPs) called `hypso1-qgis-gcps`. The GCPs were manually generated and uploaded to `hypso1-qgis-gcps` to support adding geospatial information to HYPISO-1 captures used in the datasets. Finally, calibration data and Python code was reused from the `cal-char-corr` software repository. Some parts of the calibration code were integrated into the `hsts` package in order to simplify and automate calls to the calibration code.

In addition to Python, a geographic information system (GIS) program called QGIS was also used in this project [70]. QGIS provided a manual georeferencing tool to generate sets of ground control points (GCPs) for HYPISO-1 captures as well as a graphical environment for displaying and examining processed datasets [71]. These QGIS features were used to supplement the Python code and for visualization and analysis purposes. QGIS was used in a limited capacity for plotting some of the validation data.

3.3 HYPISO-1 Data

3.3.1 Overview

A variety of HYPISO-1 capture data from different geographic sites was selected to form the time series datasets. These sites, spread across locations in Europe and North America, were selected based on the amount of HYPISO-1 captures available, number of different dates, and the occurrence or presence of algal blooms or other coastal phenomena known to be common in the region. The following geographic sites were used to generate time series datasets: the Danube River delta on the border between Romania and Ukraine, Lake Erie on the border between the US and Canada, Frohavet off the coast of Norway, and the Salish Sea on the border between the US and Canada. The locations are shown on a map in Figure 3.1.

The HYPISO-1 capture data is stored as hyperspectral datacubes that are contained in a binary file format known as "band interleaved by pixel" (BIP). These binary files use the file extension ".bip" [72]. The BIP files do not contain geospatial information about the hyperspectral data, meaning that the geospatial information must be added at a later stage of processing. These data files were obtained directly from an internal NTNU Small Satellite Lab repository of captures from HYPISO-1.

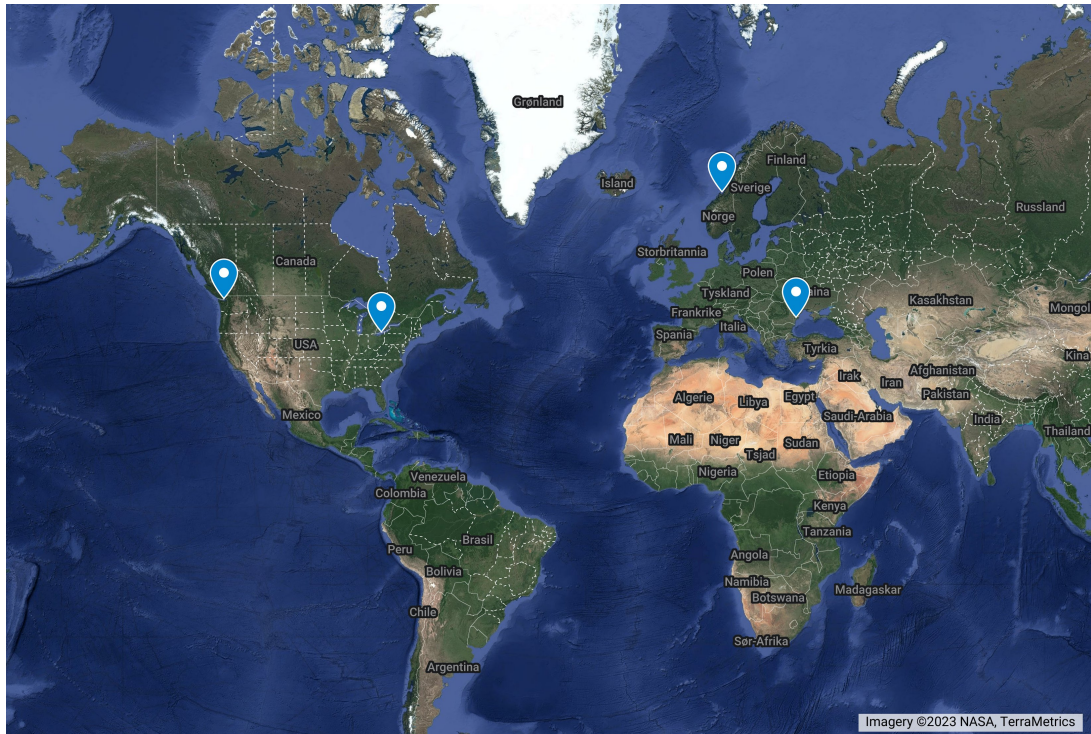


Figure 3.1: Locations of the HYPSON-1 observation targets used in this project.

3.3.2 Observation Times

The HYPSON-1 captures used in the time series datasets were primarily selected based on the amount of clouds visible in the visible (RGB) images of the captures. The datasets were created from HYPSON-1 captures that were free or mostly free of cloud cover. Captures with minimal cloud cover provided the best viewing conditions of ocean or lake regions, a factor important for change detection and ICA. Additionally, some of the captures were selected based on the season they were taken. In sites like Lake Erie, algal blooms vary based on the season and typically occur during the warmer months of the year (northern hemisphere summer) [19].

3.3.3 Lake Erie

Lake Erie, one of the Great Lakes of North America, is situated on the border between the United States and Canada. Lake Erie is a freshwater lake and is characterized by its proximity to urban areas and agricultural land. Because of its geographical location and being the outflow basin of rivers such as the Maumee River, Lake Erie is susceptible to excessive nutrient loading, made more severe by agricultural fertilizer use. The influx of nutrients encourages the growth of species of algae in the lake. Depending on the season and weather conditions, the algae growth can develop into harmful algal blooms (HABs), negatively impacting the water quality of the region. One particularly affected area is the western Lake Erie basin, which receives direct run-off from the Maumee River. [73]

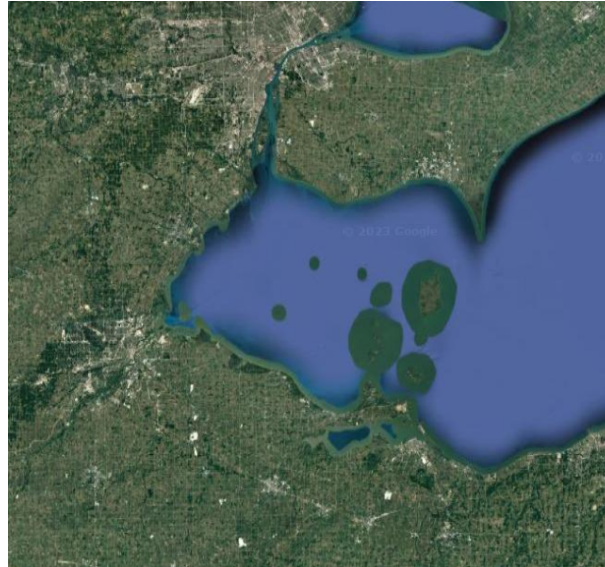


Figure 3.2: Google Maps image of the western Lake Erie basin in the Great Lakes complex of North America.

Because of its role as a major water source of residents in the region, Lake Erie is the subject of studies and monitoring campaigns. In the US, the National Oceanic and Atmospheric Administration (NOAA) produces algal bloom forecasts for Lake Erie. The forecasts are generated using a combination of Sentinel-3 satellite imagery, in-situ measurements, and water current computer models. [74]



Figure 3.3: Google Maps image of Lake Erie in the Great Lakes complex of North America. A NOAA HAB chlorophyll-a forecast product from 19 July 2022. The forecast data is derived from Sentinel 3 imagery and is overlaid on the map. [74]

For this project, the Lake Erie site was selected because of the availability of HYPSON-1 captures taken during the northern hemisphere summer in 2022, at the height of the algal bloom season in Lake Erie. Additionally, Lake Erie is well studied and monitored, making the identification of algal blooms possible through the use of the European Space Agency's Sentinel-3 satellite data and National Oceanic and Atmospheric Administration's (NOAA) HAB forecast products serving as validation data for qualitative comparison purposes. The Sentinel-3 data was made available as separate chlorophyll concentration (Chl-a) and total suspended matter (TSM) concentration data products. An example of the chlorophyll concentration product is shown in Figure 3.3. The NOAA HAB prod-

ucts are produced from measurements taken by the Ocean and Land Colour Instrument (OLCI) on-board the Sentinel 3 satellites [75]. The NOAA HAB forecast products are discussed in further detail in Section 3.7.

The Lake Erie dataset was processed from six separate HYPSON-1 captures. The capture dates and filenames are listed in Table 3.1.

Table 3.1: List of captures used for the Lake Erie dataset.

Lake Erie Dataset	
Date	HYPSON-1 Capture Filename
19 July 2022 15:50 UTC	erie_2022-07-19_1550Z.bip
20 July 2022 15:39 UTC	erie_2022-07-20_1539Z.bip
29 July 2022 15:28 UTC	erie_2022-07-29_1528Z.bip
27 August 2022 16:05 UTC	erie_2022-08-27_1605Z.bip
04 March 2023 15:17 UTC	erie_2023-03-04_1517Z.bip
07 March 2023 16:09 UTC	erie_2023-03-07_1609Z.bip

For running the ICA-based change detection techniques, a bitemporal pair consisting of two captures was selected from the Lake Erie dataset. The capture dates and filenames for the bitemporal pair are listed in Table 3.2. The RGB images of the captures are shown in Figure 3.4.

Table 3.2: List of captures used for the bitemporal Lake Erie dataset.

Bitemporal Lake Erie Dataset	
Date	HYPSON-1 Capture Filename
19 July 2022 15:50 UTC	erie_2022-07-19_1550Z.bip
27 August 2022 16:05 UTC	erie_2022-08-27_1605Z.bip

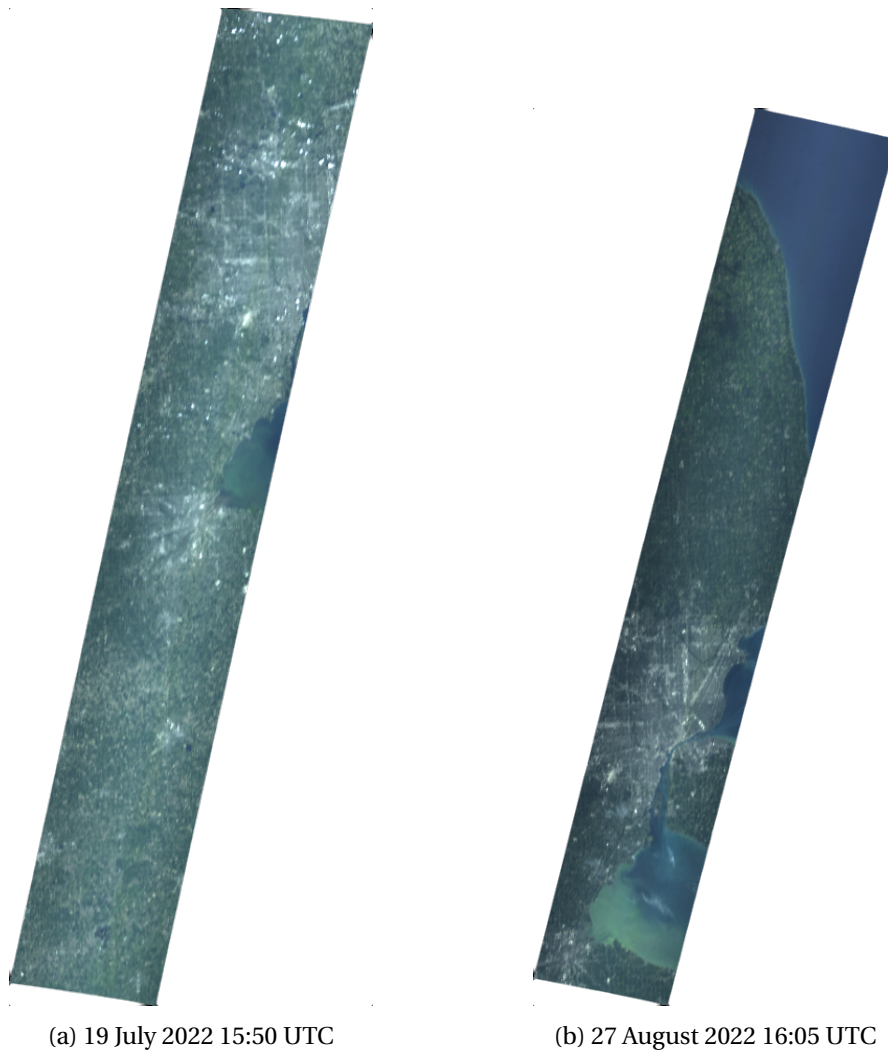


Figure 3.4: HYPSON-1 captures of Lake Erie in RGB true color.

3.3.4 Salish Sea

The Salish Sea is a body of water connected to the Pacific Ocean and located in the Pacific Northwest of North America. It spans across the coastal regions along the border of the United States and Canada and encompasses parts of British Columbia and Washington state. The major urban center of Vancouver, British Columbia is positioned along the Salish Sea along with the mouth of the Fraser River. The Salish Sea features numerous waterways including several straits and fjord-like areas containing fisheries and other aquaculture assets. The presence of these commercial facilities highlights the importance of monitoring algal blooms in the area and was the primary motivating factor in selecting the Salish Sea as one of the locations used in this project. Unlike Lake Erie, there are no dedicated NOAA HAB forecast products however Sentinel-3 OLCI products were available to be used as validation data. [75]

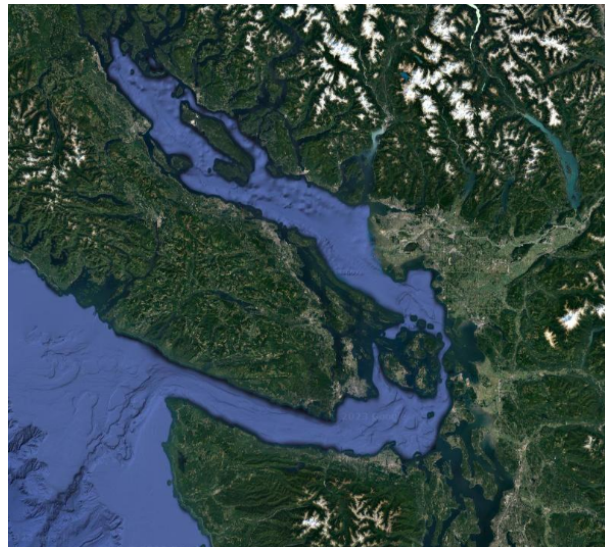


Figure 3.5: Google Maps image of the Salish Sea situated in the Pacific Northwest region of North America.

Table 3.3 lists the three separate HYPSON-1 captures included in the Salish Sea dataset, along with their respective capture dates and filenames. All of the captures were taken during the span of three days in July 2022. During the processing of the Salish Sea dataset, the capture target name was "griegBC" shortened from Grieg, British Columbia. This name was taken from the name of fisheries located within the region.

Table 3.3: List of captures used for the Salish Sea dataset.

Salish Sea Dataset	
Date	HYPSON-1 Capture Filename
12 July 2022 18:46 UTC	griegBC_2022-07-12_1846Z.bip
13 July 2022 18:34 UTC	griegBC_2022-07-13_1834Z.bip
14 July 2022 18:22 UTC	griegBC_2022-07-14_1822Z.bip

For running the ICA-based change detection techniques, a bitemporal pair consisting of two captures was selected from the Salish Sea dataset. The capture dates and filenames for the bitemporal pair are listed in Table 3.4. The RGB images of the captures are shown in Figure 3.6.

Table 3.4: List of captures used for the bitemporal Salish Sea dataset.

Bitemporal Salish Sea Dataset	
Date	HYPSON-1 Capture Filename
12 July 2022 18:46 UTC	griegBC_2022-07-12_1846Z.bip
13 July 2022 18:34 UTC	griegBC_2022-07-13_1834Z.bip

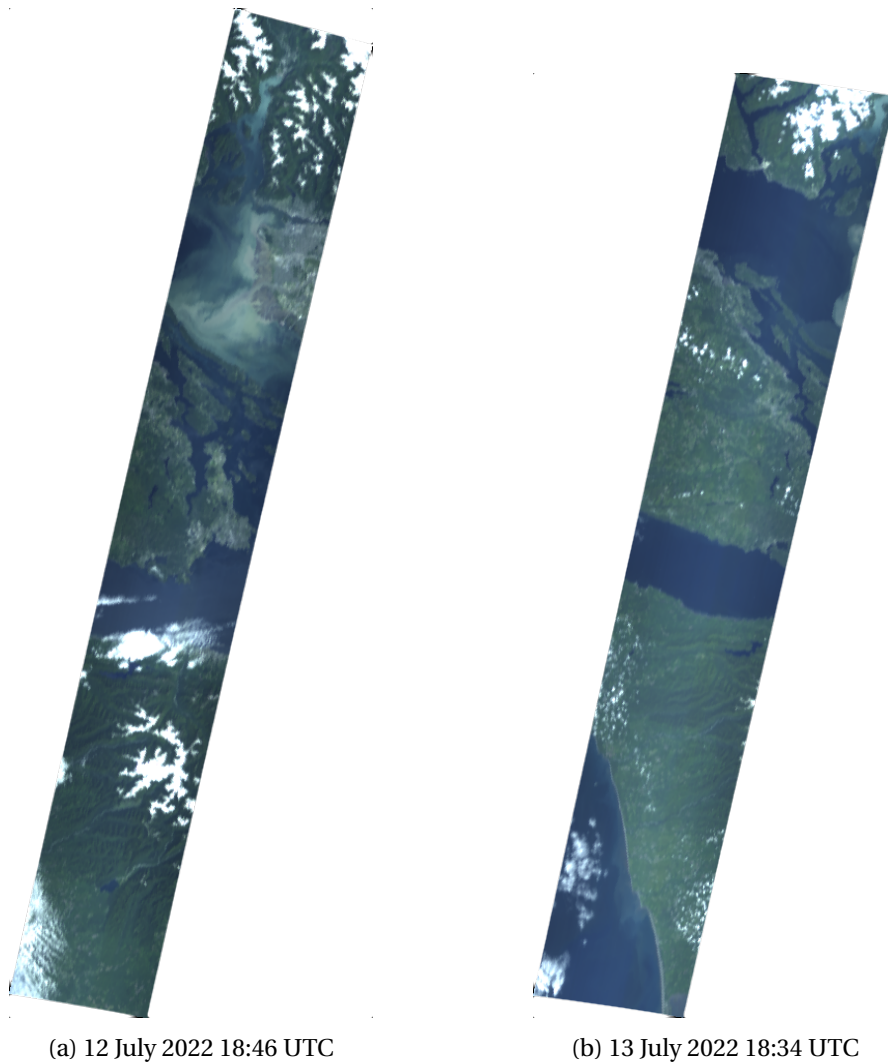


Figure 3.6: HYPSON-1 captures of the Salish Sea in RGB true color.

3.3.5 Danube River Delta

The Danube River Delta is a region located at the confluence of the Danube River and the Black Sea in Southeastern Europe along the border of Ukraine and Romania. The area is composed of numerous river distributaries, lakes, and wetlands. Because of its unique geography with many different bodies of water and location on the Black Sea, it was selected as one of the observational sites of HYPSON-1. One of the major features of the site is a large fresh water lagoon called Lake Razim located directly along the southern extent of the delta in Romania, shown in Figure 3.8. Lake Razim is connected to a complex of smaller lakes called Lake Golovita and Lake Zmeica. Adjacent and south of these lakes is Lake Sinoe. These four lakes provide a distinct point of comparison against the salt water in the Black Sea and are heavily influenced by human activity in the region including from agriculture, irrigation, and pollution. Seasonal algal growth has also been confirmed in Lake Razim [76]. For the Danube River Delta site, Sentinel-3 OLCI products are available as validation data. The Danube River Delta dataset consists of three overlapping images near the Danube River delta in the Black Sea.

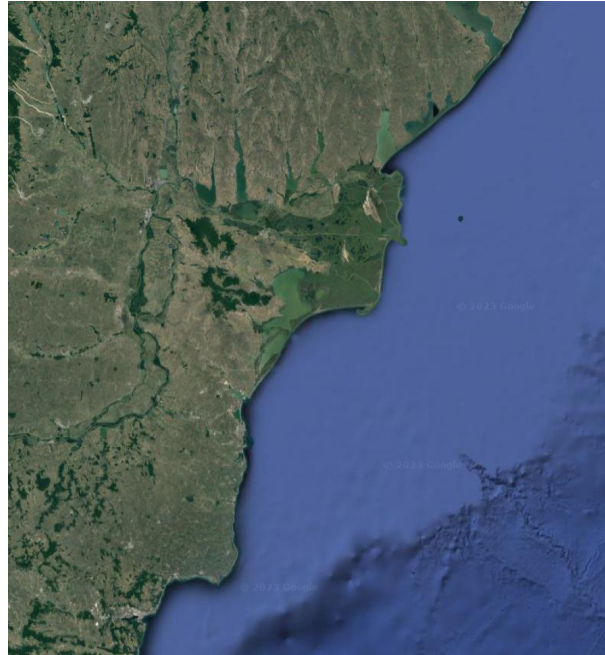


Figure 3.7: Google Maps image of Lake Razim and the Danube River Delta located on the coast of the Black Sea. Lake Razim is positioned in the center of the image.



Figure 3.8: Annotated map of the lakes near the Danube River Delta.

Table 3.5 lists the three HYPSON-1 captures included in the Danube River Delta dataset, along with their respective capture dates and filenames. During the processing of the Danube River Delta dataset, the chosen capture target name was "gloria," which corresponds to an offshore ocean color calibration site located within the area [77].

Table 3.5: List of captures used for the Danube River Delta dataset.

Danube River Delta Dataset	
Date	HYPSON-1 Capture Filename
13 February 2023 08:35 UTC	gloria_2023-02-13_0835Z.bip
05 March 2023 08:41 UTC	gloria_2023-03-05_0841Z.bip
08 April 2023 08:22 UTC	gloria_2023-04-08_0822Z.bip

For running the ICA-based change detection techniques, a bitemporal pair consisting of two captures was selected from the Danube River Delta dataset. The capture dates and filenames for the bitemporal pair are listed in Table 3.6. The RGB images of the captures are shown in Figure 3.9.

Table 3.6: List of captures used for the bitemporal Danube River Delta dataset.

Bitemporal Danube River Delta Dataset	
Date	HYPSON-1 Capture Filename
05 March 2023 08:41 UTC	gloria_2023-03-05_0841Z.bip
08 April 2023 08:22 UTC	gloria_2023-04-08_0822Z.bip

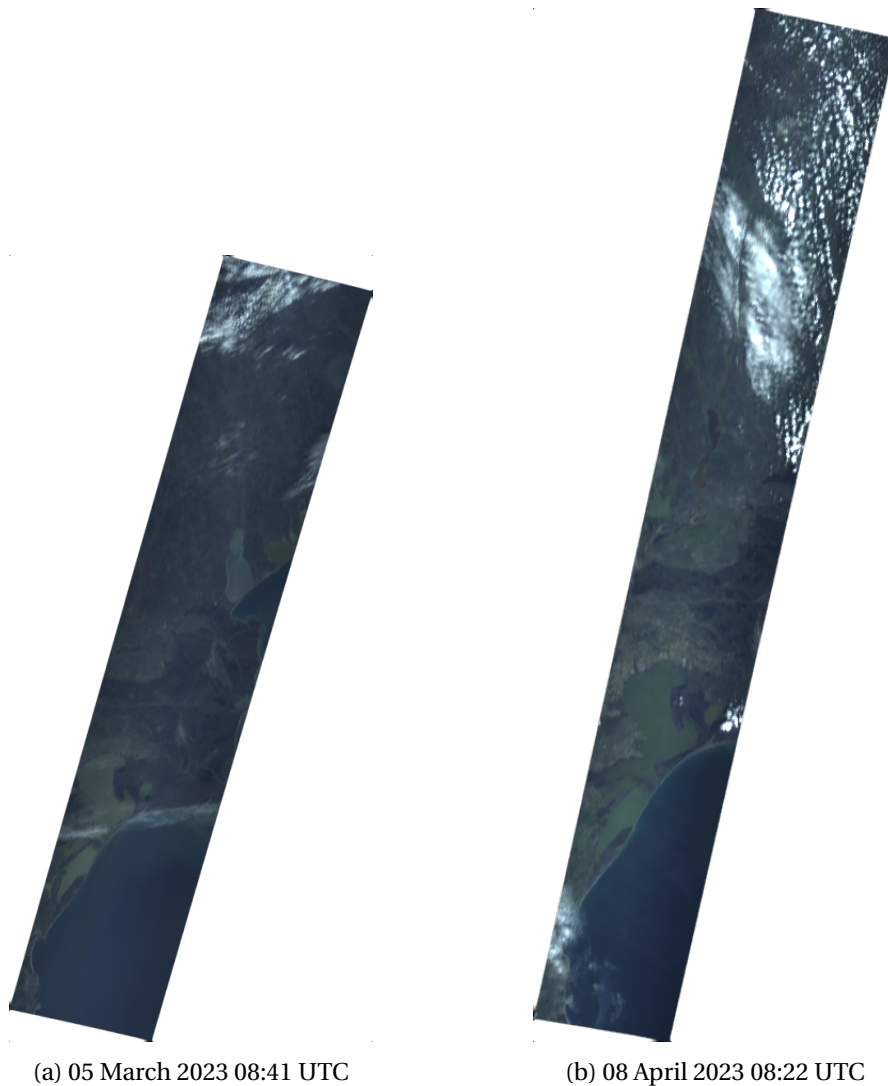


Figure 3.9: HYPSON-1 captures of the Danube River Delta in RGB true color.

3.3.6 Frohavet

The final observation site that was chosen was Frohavet, a coastal region along the Norwegian coastline in Northern Europe. The region encompasses a wide range of geographic features including fjords, archipelagos, and open sea areas, as seen in Figure 3.10. These varied features are good for evaluating the performance of the ICA-based change detection techniques and land mask in narrow channels as well as in open water. Trondheim Fjord is a major geographic feature in the area and extends into the interior of Norway. Owing to its proximity to Trondheim and NTNU, the Frohavet is the most familiar and studied observation site. The close location makes the region a prime candidate in the future for potential in-situ water sampling or observations with other mobile assets such as boats or drones. These activities would be performed with the intent to develop an observational pyramid for oceans, incorporating a hyperspectral satellite such as HYPSON-1 [1]. Ongoing research in the region also opens the possibility for future collaboration.



Figure 3.10: Google Maps image of Frohavet and Trondheim Fjord.

The dataset for Frohavet included six HYPSON-1 captures. The captures were taken during periods in November 2022 and March 2023, providing hyperspectral data of the same area from two different seasons. The HYPSON-1 captures for Frohavet are listed in Table 3.7.

Table 3.7: List of captures used for the Frohavet dataset.

Frohavet Dataset	
Date	HYPSON-1 Capture Filename
17 November 2022 10:21 UTC	frohavet_2022-11-17_1021Z.bip
18 November 2022 10:08 UTC	frohavet_2022-11-18_1008Z.bip
10 December 2022 10:12 UTC	frohavet_2022-12-10_1012Z.bip
16 March 2023 10:44 UTC	frohavet_2023-03-16_1044Z.bip
28 March 2023 10:59 UTC	frohavet_2023-03-28_1059Z.bip
29 March 2023 10:44 UTC	frohavet_2023-03-29_1044Z.bip

For running the ICA-based change detection techniques, a bitemporal pair consisting of two captures was selected from the Frohavet dataset. The capture dates and filenames for the bitemporal pair are listed in Table 3.8. The RGB images of the captures are shown in Figure 3.11.

Table 3.8: List of captures used for the bitemporal Frohavet dataset.

Bitemporal Frohavet Dataset	
Date	HYPSON-1 Capture Filename
28 March 2023 10:59 UTC	frohavet_2023-03-28_1059Z.bip
29 March 2023 10:44 UTC	frohavet_2023-03-29_1044Z.bip

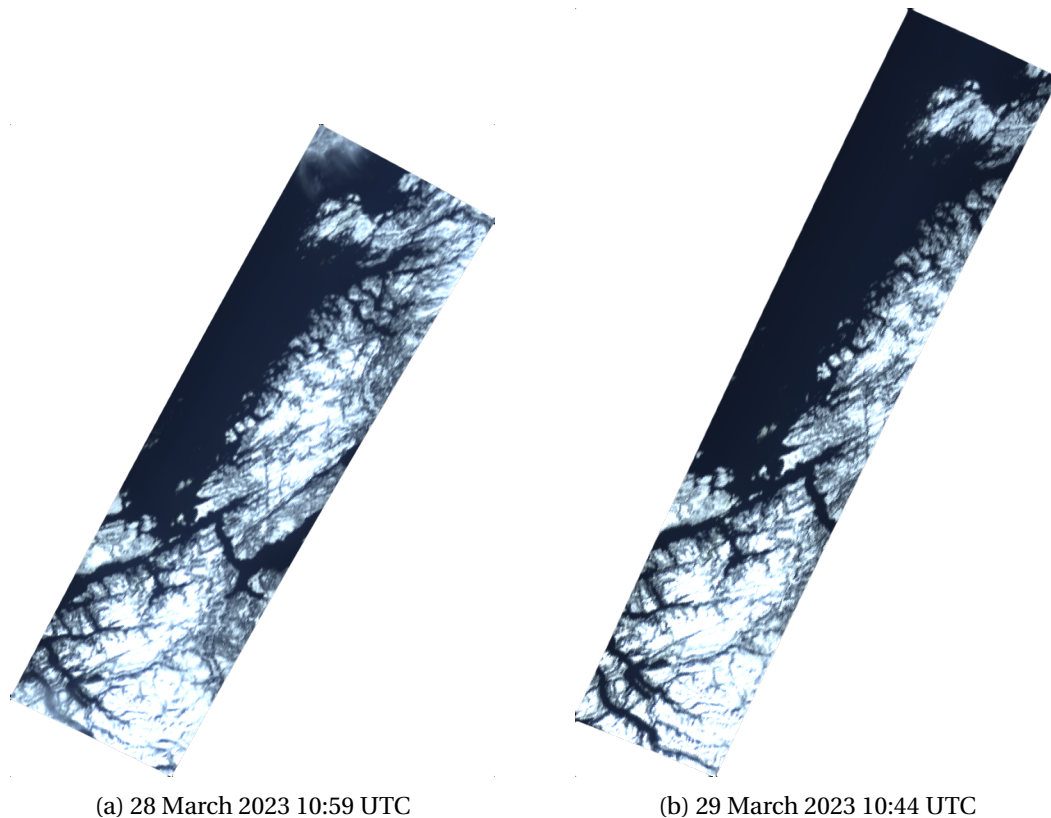


Figure 3.11: HYPSON-1 captures of Frohavet in RGB true color.

3.4 Time Series Processing

3.4.1 Overview

A data processing pipeline was created to generate multitemporal time series datasets from the raw, unprocessed HYPSON-1 data. This task required steps such as spectral and radiometric calibration of the data, georeferencing of the data, and resampling the data to the same spatial resolution. Additionally, the processing pipeline included applying a preexisting land mask to the data, simplifying the process of separating land and water regions of the captures. The following sections describe the steps involved in the data processing pipeline.

3.4.2 Loading Data

The first step of the processing pipeline was to load the raw HYPSON-1 data. The HYPSON-1 hyperspectral datacubes are encoded in .BIP binary files [78]. These files contain the raw sensor data that is uncalibrated and ungeocoded. The .BIP extension used by these binary files is an abbreviation for band-interleaved-by-pixel and describes how the data is structured and organized within the file. Each .BIP file contains exactly one HYPSON-1 datacube requiring multiple .BIP to be loaded individually in sequence in order to create a complete time series dataset. The process of reading the files involved utilizing the `np.fromfile` function from NumPy, which resulted in the creation of a three-dimensional NumPy matrix containing a datacube for each capture [79]. The resulting matrices had the dimensions of 956 by 684 by 120, the standard HYPSON-1 capture resolution. As a final step, the

cross track dimension of each capture was binned by a factor of 3 to obtain an image with the correct aspect ratio. This reduced the cross track dimension from 684 down to 228, yielding NumPy matrices with final dimensions of 956 by 228 by 120 pixels.

3.4.3 Calibration

The second stage of the dataset processing pipeline was data calibration. This stage included the important steps of spectral and radiometric calibration of the raw HYPSON-1 data. Spectral calibration aims to create a correspondence between the spectral bands in the datacube to the physical wavelengths that the bands represent. This type of calibration is vital to accurately interpret spectral patterns in the hyperspectral components. In contrast, radiometric calibration is used ensure radiance values measured by the HSI accurately represent the actual radiance values of the source. [80] [1]

During the calibration process, values in the hyperspectral datacubes were also converted from their original radiance values to reflectance values. The purpose of this conversion is to attempt to minimize and remove the effects of viewing and illumination angles (solar zenith angles) of the target. Reflectance values are more easily compared between captures than radiance values [1]. Noise removal was also performed on the data in the form of spatial destriping. The noise removal step is attempts to remove sensor artifacts introduced in the data by the push-broom scanning technique used by HYPSON-1's HSI.

The spectral and radiometric calibration of the hyperspectral data was done using calibration data from the NTNU SmallSat Lab's `cal-char-corr` internal software repository hosted on GitHub. This repository contains sets of calibration constants information to map bands in the hyperspectral to their corresponding wavelengths [81]. The `cal-char-corr` repository included a set of spectral calibration coefficients and a set of radiometric calibration coefficients, each stored in a comma separated variable (CSV) text file. The code used to perform calibration tasks was largely based on Python code from [1]. The Python code handled most of the tasks for spectral calibration, radiometric calibration, conversion from radiance to reflectance, and spatial destriping. It was used in conjunction with the most recent calibration constants from the `cal-char-corr` repository and was responsible for loading and reading the CSV calibration files. The end result of the calibration stage of the pipeline were spectral- and radiometric-calibrated NumPy matrices for each HYPSON-1 capture.

3.4.4 Georeferencing

An important aspect of satellite imagery and data is the ability to map image pixels to geographic coordinates. This allows features or objects in the images to be spatially located on Earth's surface. At the time of writing, HYPSON-1 data does not have reliable geospatial information that can be associated with the images. Direct methods of generating geospatial information from the orbit and orientation of HYPSON-1 are under development however they are currently not accurate enough for the purpose of time series dataset processing. As a work around to this shortcoming, an indirect alternative approach involving ground control points (GCPs) was utilized. GCPs are a series of defined points in images that are mapped to known geographical coordinates. By using the GCPs, geospatial information can be derived from the rest of image through a process known as georeferencing.

Georeferencing is a multi-step process. First, GCPs must be selected and generated. For the HYPSON-1 data used in this project, the associated GCPs were manually generating using the Geo-

referencer tool in QGIS [71]. Second, following the selection of GCPs, a geometric transformation must be derived from the GCPs to calculate the geospatial information (i.e. geographic coordinates) for the rest of the image. The Python libraries `skimage` [61] and `rasterio` [59] were used for this step of georeferencing.

Selecting GCPs

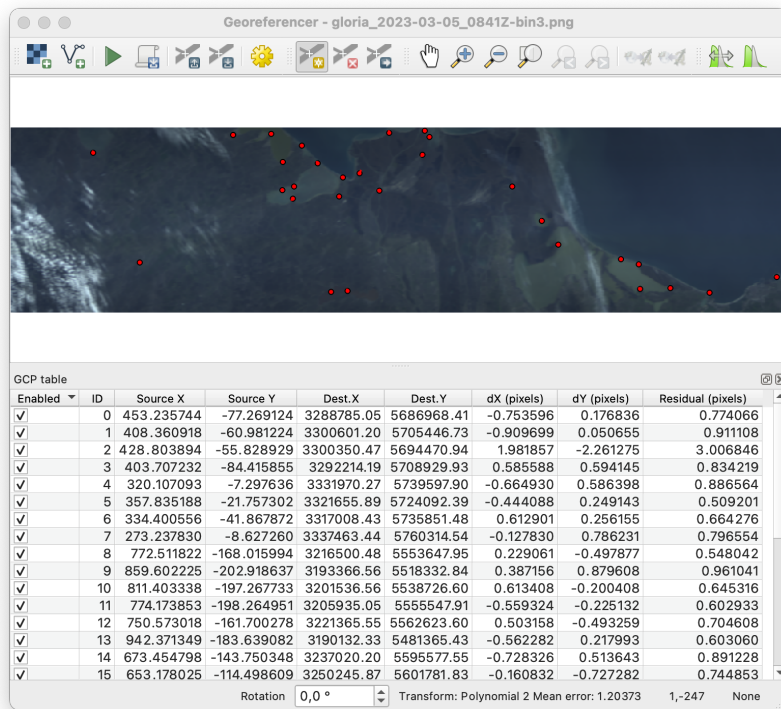
The Georeferencer tool in QGIS was used to select and generate a list of GCPs [71]. Using the Georeferencer tool, RGB images generated from the HYPISO-1 captures were compared to a map product based on Google Maps imagery. Static features in the RGB images such as mountains, islands, lakes, and rivers were used as landmarks. When a landmark was identified, a pixel coordinate close to the landmark was selected in the RGB image of the capture. The exact geographical coordinates were then obtained by selecting the corresponding point in the Google Maps imagery. Using this process, a one-to-one corresponding list of GCPs was slowly created. For each capture, a minimum of ten GCPs were selected and emphasis was made on selecting GCPs from throughout the image rather than focusing on one area. By selecting GCPs spread evenly throughout the image, the geospatial information becomes more accurate in all parts of the image. The GCPs were saved to `.points` files, one for each HYPISO-1 capture included in the datasets. The `.points` files are formatted as standard comma separated variable (CSV) files allowing the GCP data to be easily read into Python scripts and used to assign geospatial information for the rest of the pixels in the corresponding HYPISO-1 capture.

When working with GCPs, a geographic coordinate reference system (CRS) must be chosen. The CRS is used to define the map projection and coordinate system used by QGIS and Python geospatial libraries. In this project, a standard called EPSG:3857, also known as pseudo-Mercator, was used. EPSG:3857 was selected as the CRS because it handles severe mapping projection distortions near the Earth's poles better than the default EPSG:4326 CRS used in QGIS. The `pyproj Transformer` Python library was used in the dataset pipeline code for converting CRSs as well as a way to add support for additional CRSs in the code in the future. [60]

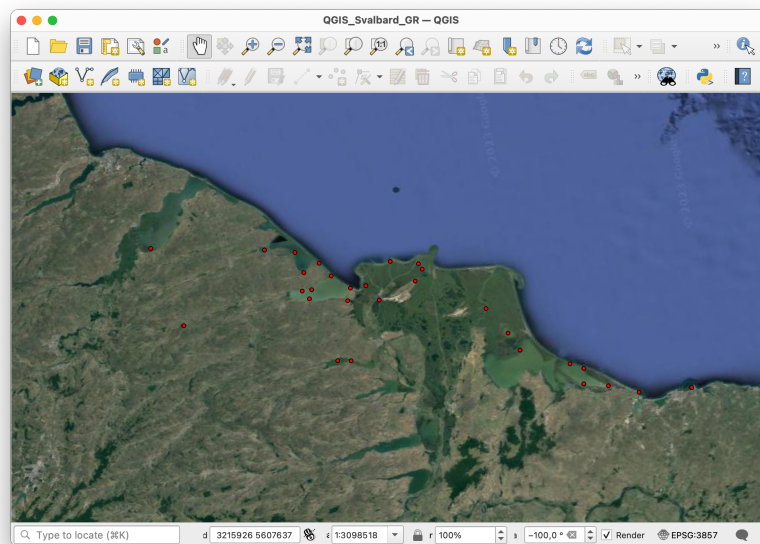
Geometric Transformation

After generating the GCPs for each capture using the Georeferencer tool in QGIS, the GCPs were used to calculate geographic coordinates for the rest of the pixels in the captures. In this step, the HYPISO-1 captures were treated as raster data, a type of data that consists of a grid of cells (i.e. pixels). The GCPs were used to align the cell data to a predefined geospatial coordinate system in a process known as a geometric transformation. The result of this process is that each cell or pixel in the raster data will have an associated geographic coordinate. The latitude and longitude values can be used later for resampling the captures and constructing a time series dataset from multiple spatially overlapping captures.

There are multiple software approaches for performing geometric transforms. QGIS has built-in support for geometric transforms using GCPs as input however QGIS only supports transformations for single channel or RGB image data and not hyperspectral datacubes. The inability of QGIS to transform hyperspectral data ruled out using QGIS for performing geometric transform in this project. Another common software tool that provides geometric transforms is the Python library `Rasterio` [59]. Using a set of GCPs and the `Rasterio` functions



(a) QGIS georeferencing tool being used to add GCPs to a HYPISO-1 capture of the Danube River delta. The GCP table displays a list of pixel coordinates and the corresponding EPSG:3857 geospatial coordinates.



(b) GCPs overlaid on Google Maps reference imagery.

Figure 3.12: QGIS georeferencing tool.

`rasterio.transform.from_gcps` and `rasterio.transform.xy`, hyperspectral data can be geometrically transformed and georeferenced. One of the major issues with Rasterio, however, is that Rasterio only supports a single type of transformation known as an affine transformation. Affine transforms preserve lines and parallel features in images but cannot handle non-linear distortions. In the case of HYPISO-1 imagery, non-linear distortions are present due to a combination of viewing angles, movement of the sensor, and the Earth's curved geometry. As a result, some HYPISO-1 captures processed using an affine transformation display severe distortions which reduce spatial accuracy of the data.

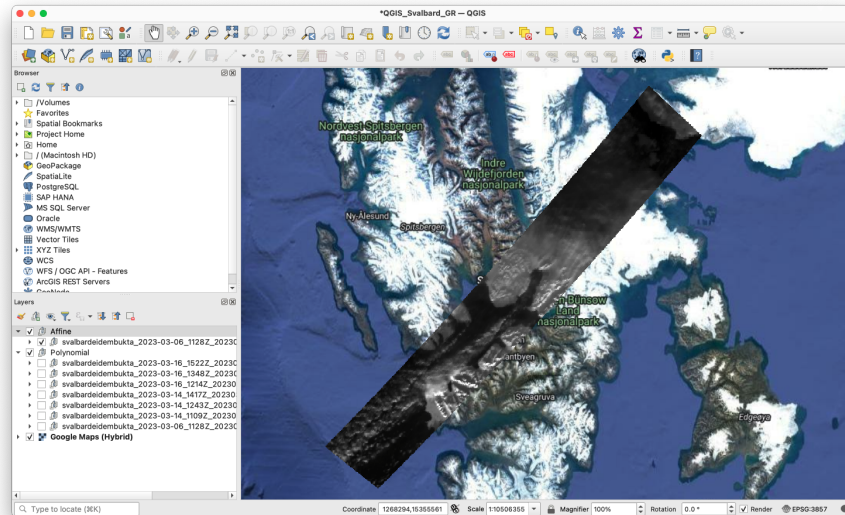
To better handle non-linear distortions in the captures, the Rasterio-based affine transformation was replaced with a custom developed polynomial transformation approach. In the polynomial transform approach, pixel and geographic coordinate pairs were used to estimate the coefficients for second-order multivariate polynomials used to calculate latitude and longitude coordinate values. The polynomials used pixel indices used as inputs and could be fit to the non-linear distortions in the images.

The polynomial transformation approach was implemented in the software using the `skimage` Python library. This library was utilized to estimate the coefficients for the latitude and longitude multivariate polynomials. For each HYPISO-1 capture, the polynomials were fit to image and geographic coordinate data from the capture's GCPs using the `skimage.transform.estimate_transform` function [82]. The resulting coefficients corresponded to second-order multivariate polynomial equations. Since GCPs are specific to each capture, unique sets of polynomial coefficients were obtained. By placing the coefficients into their respective positions, transformation equations were derived for each capture. These equations enabled the accurate conversion of image coordinates (x and y) to geographic coordinates (latitude and longitude). The calculated latitude and longitude values for each capture were then written into NumPy matrices. Following their calculation, the latitude and longitude NumPy matrices were incorporated into the hyperspectral data by converting and combining the data into a `GeoDataFrame` object using the `Pandas` Python library. The `GeoDataFrame` object provided efficient data storage and querying capabilities while maintaining spatial information. Within the `GeoDataFrames`, the data from each spectral band, latitude values, and longitude values were stored in separate `DataSeries` data structures. The Python code written for the polynomial transformations was integrated into the `hsts` Python package.

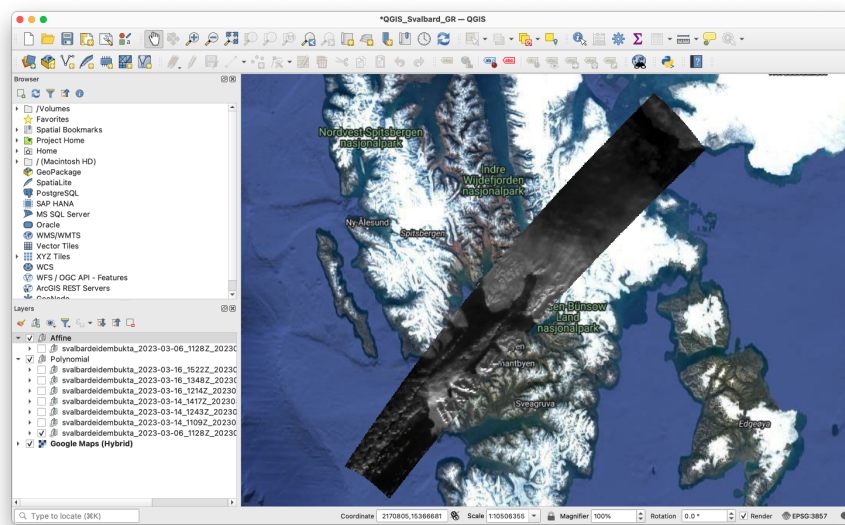
3.4.5 Resampling and Interpolation

Following the georeferencing and transformation of the data, the spatial resolution of the captures can deviate from the original 956 pixels by 684 pixels (or 956 pixels by 684 pixels, if binned) resolution of HYPISO-1 captures [13]. When a geometric polynomial transformation is applied to raster data, the original pixel locations and sizes change, resulting in a mismatch between the transformed data and the desired grid of pixels. Because each capture is transformed different set of polynomial transforms, the captures do not share a common grid of pixels. As a consequence, it becomes necessary to apply a resampling procedure to the data to ensure that the transformed raster data aligns properly to a common grid or pixel resolution shared across all of the captures.

Resampling consists of calculating new pixel values for the transformed raster data using the original data. The overall goal of resampling is to maintain an accurate and complete representation of



(a) Single channel capture image transformed using an affine transformation.



(b) Single channel capture image transformed using a second order polynomial transformation. Non-linearity is accurately preserved at the bottom or southern-most extent of the capture.

Figure 3.13: Visual comparison of an image transformed using affine and polynomial transformations.

the data while obtaining a common spatial resolution for all the captures. The resampling process changes the spatial resolution and pixel sizes of the raster data and results in a different arrangement of pixels for the data. In some cases, it is necessary to calculate values at new locations falling between the resampled pixels through the use of interpolation. This is often necessary when resampling raster data to a higher resolution than the original data. Interpolation fills in the gaps between the new pixel locations by estimating values based on neighboring pixels. Applying interpolation to resampled raster data ensures that there are no gaps or holes present in the data.

Resampling

For this project, resampling was implemented using a rudimentary combination of the GeoCube [62], xarray [55], and SciPy [54] Python libraries. First, the xarray datasets were iterated through to identify the most extreme latitude and longitude values present in the data. These values were used to create a spatial bounding box ("bbox") and define a common spatial resolution grid to be shared between captures in the time series dataset. The x axis and y axis spatial resolutions of individual resampled pixels was set to 0.006 degrees, the default value used by the GeoCube Python library.

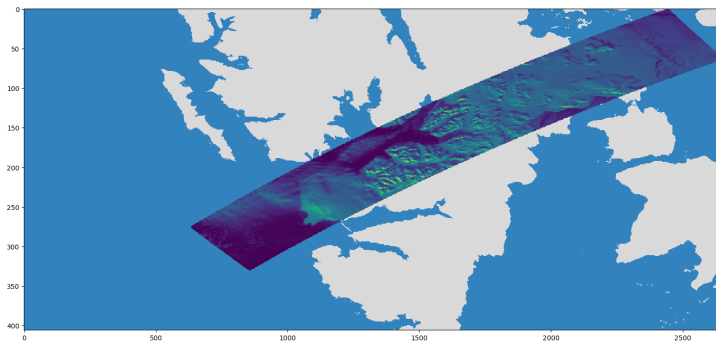
Next, after computing a common resolution and grid for the dataset, the data from each capture was resampled using the `make_geocube` function from the GeoCube library [62]. The resampling process was carried out individually for each spectral band from a given capture. Information about the desired resolution and geometry of the resampled data was passed as arguments to the `make_geocube` function. Since HYPSON-1 captures have 120 bands, the function was called 120 times to resample each band separately. The output of each call of the `make_geocube` function was a rasterized xarray series with x and y dimensions equal to that of the common spatial resolution grid. Each of these xarray series corresponded to a separate spectral band from the capture.

Figure 3.14 shows an example of three different single channel images resampled to a common spatial resolution as is done GeoCube and `make_geocube`. A side effect of the resampling process is that the rasterized capture data will be surrounded by NaN (Not-a-Number) values within a spatial resolution grid that extends beyond the rasterized capture data. This is due to the fact that the common spatial resolution grid is defined using the largest and smallest latitude and longitude values from all of the captures, not just individual ones. For individual captures, many of the pixel locations fall outside the extent of the original image after being resampled and aligned to the common grid. These pixel locations are undefined and are handled by assigning them a NaN value due to the lack of original data that could be used to assign a value to them.

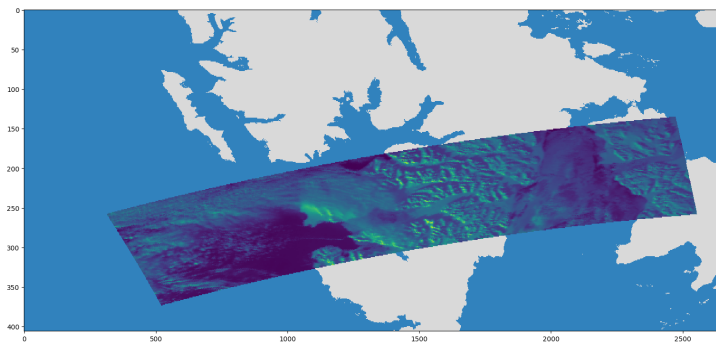
Interpolation

An interpolation procedure was developed using functions from the SciPy [54] library. The purpose of the interpolation procedure was to fill any gaps or holes in the resampled raster data. This was divided into two distinct steps. First, NaN values in the resampled raster data were located and identified. These NaN values indicated areas of the data that needed to be filled using interpolation. Following the identification of missing data, a second step consisting of linear interpolation was used to compute values for the missing data using information from neighboring pixels.

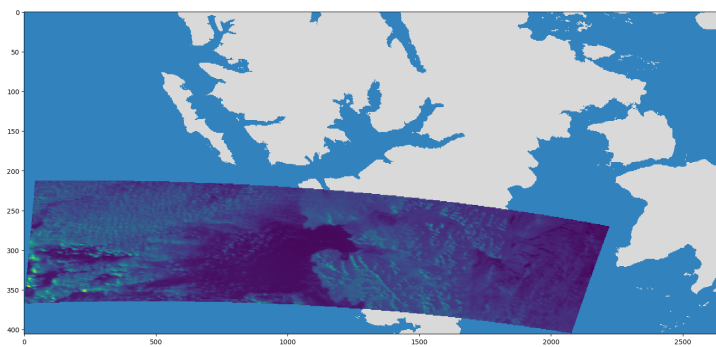
A method called binary closing was used to find missing pixels. Binary closing is a mathematical operation that used to combination of dilation and erosion operations to find and fill holes in binary



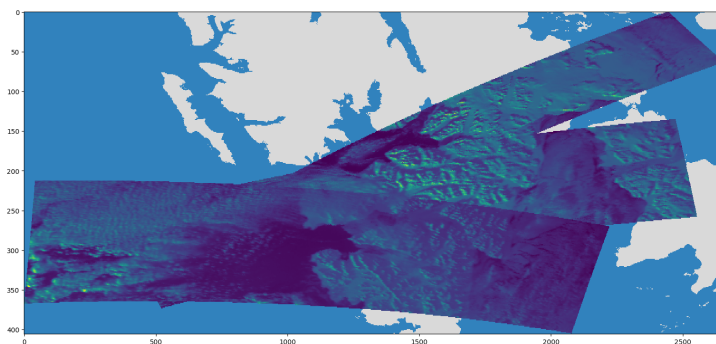
(a) Resampled single channel capture 1.



(b) Resampled single channel capture 2.



(c) Resampled single channel capture 3.



(d) Resampled single channel captures combined and overlaid in the same image.

Figure 3.14: Examples of resampled single channel captures from the same dataset.

data. In each spectral band raster, the data was treated as binary data, that is, it was either marked as valid or missing. A Python function from the SciPy library called `scipy.ndimage.binary_closing` and a closing size of three pixels was used to perform the binary closing operation [83]. The output of the binary closing operation was a mask indicating pixel locations that needed to be assigned values through interpolation. One of the strengths of the binary closing method is the ability to distinguish between small, localized gaps of missing data from the expected NaN values filling the raster grid outside the edges of the capture data. Alternative methods such as convex hull operations fail to make this important distinction and consequently make false positives on pixel locations not requiring interpolation. For the interpolation step, SciPy was utilized once again. The function `scipy.interpolate.griddata` was used to interpolate the points using the linear interpolation mode [84]. Once values for missing points were calculated using linear interpolation, they were used to replace the NaN values in the raster data.

Similar to the resampling procedure, the binary closing and interpolation steps were iteratively applied to each spectral band separately and outputted a distinct xarray series for each band. Afterward, the collections of xarray series, representing each capture, were combined to form unified xarray datasets. These datasets brought together the georeferenced and resampled data for each HYPSON-1 capture. The consolidation ensured that every capture in the time series dataset had a single data structure containing all the relevant spectral and geospatial values associated with it.

3.4.6 Land Mask

For an ocean monitoring satellite such as HYPSON-1, the ability to separate land and water areas within a dataset is crucial. By distinguishing between land and water, change detection analysis can focus exclusively on properties and features in water while disregarding land-based signals and influences. To minimize the time and work needed to distinguish between land and water pixels in HYPSON-1 hyperspectral data, binary land masks were added to the datasets. A binary land mask is a spatial map of land and water areas in a raster format, where each pixel is assigned a binary value of true (1) or false (0). Land pixels are indicated with true values, while water pixels are indicated with a false value. The binary or boolean nature of land masks allow for non-water pixels to be removed easily during analysis using simple logical operations.

There are different methods of creating land masks for satellite data. One method is to use the satellite data itself and attempt to classify pixels as either land or water. This can be achieved using supervised classification algorithms such as support vector machines (SVMs). An alternative approach is to use predefined land masks created using an auxiliary data source. It is this second approach that was used for creating land masks for the HYPSON-1 time series datasets.

The data source for the land masks was the Global Self-consistent, Hierarchical, High-resolution Geography Database or GSHHG [85]. The GSHHG database is a widely used collection of geographic and shoreline ESRI shapefiles compiled from public sources. GSHHG has global coverage and includes shapefiles that detail ocean, lake, river, and island shorelines. GSHHG is also available at different resolutions ranging from "full" to "crude". To construct the land mask for the time series datasets, the full resolution GSHHG data was used. GSHHG consists of a series of ESRI shapefiles of various shoreline hierarchies ranging from islands located in inland bodies of water. To simplify using GSHHG, the different hierarchical shapefiles were all combined and saved using QGIS to create a single consolidated global land mask shapefile. The consolidated shapefile included the maximum

number of shoreline hierarchies available from the GSHHG.

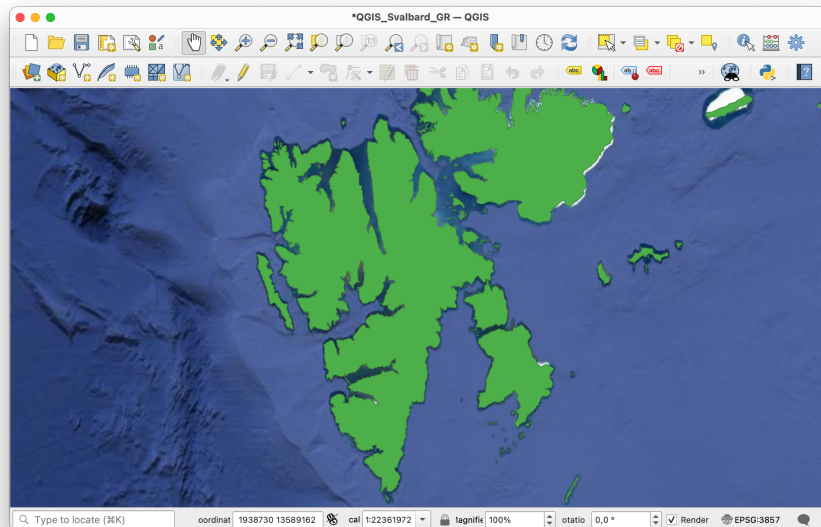


Figure 3.15: Land mask covering Svalbard.

In order to add the land mask to the HYPSON-1 time series dataset, it was necessary to load and open the consolidated GSHHG shapefile. This was achieved using the Fiona Python library which produced a `fiona.collection.Collection` object consisting of a list of spatial vector features from the GSHHG shapefile [64]. Since the GSHHG has global coverage, the collection of spatial vector features loaded by Fiona was too large to be used directly and added to the hyperspectral capture data. Instead, the spatial vector collection was filtered by searching for any spatial intersections with the bounding box generated during the resampling step. Recall that the bounding box was defined by the most extreme latitude and longitude values present in the data and effectively described a spatial region of interest for the dataset. This approach of filtering the data extracted a list of land mask polygons for areas within the bounding box while discarding unneeded land mask data falling outside of it. The query for spatial intersections was done using the Shapely Python library [63].

The final step in generating the land mask was to combine the list of land mask polygons and rasterize them to the same resolutions as the resampled hyperspectral data. The Rasterio library was a major component in the implementation [59]. The rasterization of the land mask data was implemented using the `rasterio.features.geometry_mask` function. Several arguments were required, including the list of polygons to be combined, the desired height and width of the final rasterized land mask (values reused from the resampled hyperspectral data), and a transform derived from the bounding box using the `rasterio.transform.from_bounds` function. The output of this procedure was a boolean NumPy matrix that was then added as an additional land mask variable in the xarray datasets containing the resampled hyperspectral data.

Overall, the described process of generating a land mask for the hyperspectral data was efficient enough for the purposes of this project. The process only needed to be repeated once per dataset since the each resampled capture in the dataset shared the same spatial resolution. The primary drawback with the land mask approach is the accuracy of the GSHHG database. Since the GSHHG

database is a compilation of different shoreline sources, it does not reflect recent changes to shorelines, either caused by physical changes or changes in mapping practices. This was noticeable at a Svalbard site that was processed with the Python code where changes in sea ice and receding glaciers have resulted in discrepancies between the apparent shoreline in satellite images and the shoreline reflected in the GSHHG dataset. In the future, robust algorithms such as SVM classifiers capable of distinguishing between water and land pixel in hyperspectral data may improve on the current land masking technique.

3.4.7 Cloud Mask

One step that was excluded from the dataset processing procedure was a cloud mask generation process. The motivating factor was primarily accuracy and reliability of the cloud mask. In previous work on ICA anomaly detection [3], a cloud mask was generated for each capture using a threshold method based on using radiance values near 640 nm, a band commonly used for daytime cloud detection [86]. The threshold method did not always perform well and frequently produced false positives in captures featuring regions with visible sediments in water or snow and ice. Additionally, each capture required a different threshold value to be set manually, a process that becomes time consuming for a dataset consisting of multiple captures. These problems along with the sufficient availability of cloud-free captures for this project were the main motivations to exclude a cloud mask from the dataset generation procedure. That being said, the addition of a reliable cloud mask in the future would still be an improvement and could potentially improve the performance of ICA. SVM-based classifier algorithms are a potential candidate for generating cloud masks for hyperspectral data.

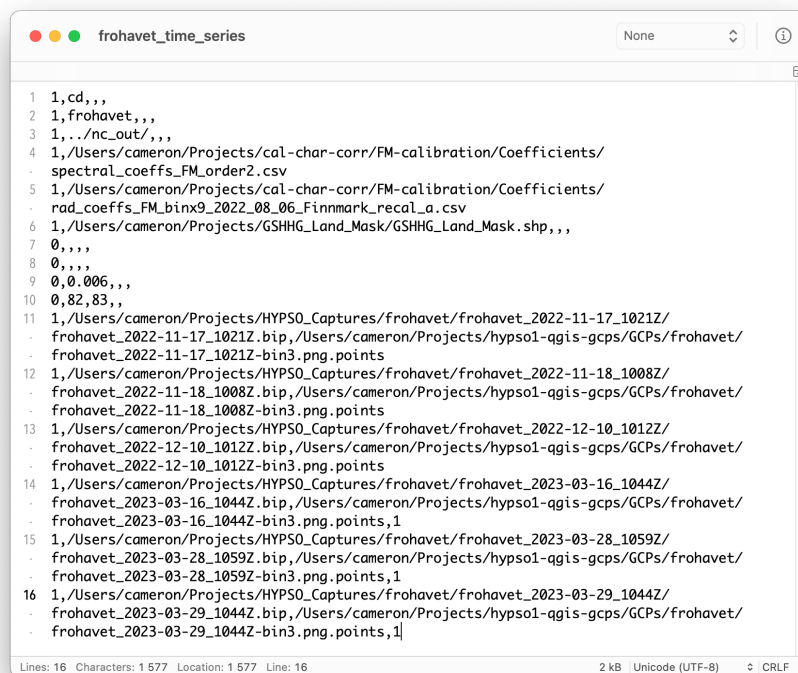
3.4.8 Writing to NetCDF

As a final step in the dataset processing, the xarray datasets were written and saved as NetCDF files, with each file containing data from exactly one HYPSON-1 capture. Conversion to NetCDF was done using the built in `xarray.Dataset.to_netcdf` function. The NetCDF files were written to separate directories corresponding to the dataset to which they belonged to. These files were later loaded for use in the ICA change detection algorithms. Various pieces of metadata were also written to the NetCDF files. This metadata included processing details, filenames, code versions, and other relevant information. Additionally, an array containing the physical wavelength labels for the spectral bands was written to the metadata. These labels were calculated from information provided by the spectral calibration files and assist with interpreting and presenting the hyperspectral data.

3.4.9 Input File

During the generation of hyperspectral time series datasets, text-based input files were used to organize and simplify the management of dataset processing settings. There were several motivations for the use of input files. Firstly, input files enhance readability and organization of processing settings. By representing the settings in a text format, they can be easily documented and saved for future reference. Additionally, it adds flexibility to dataset processing workflows. The text file can contain various processing configurations, allowing for different settings to be modified without altering the underlying dataset processing Python code. Furthermore, input files enable automation and reproducibility when generating datasets. By utilizing a standardized input file format, the dataset process-

ing pipeline can be automated, ensuring consistency and the accurate replication of results across different runs.



```

1 1,cd,,,
2 1,frohavet,,,
3 1,..nc_out/,,,
4 1,/Users/cameron/Projects/cal-char-corr/FM-calibration/Coefficients/
  spectral_coeffs_FM_order2.csv
5 1,/Users/cameron/Projects/cal-char-corr/FM-calibration/Coefficients/
  rad_coeffs_FM_binx9_2022_08_06_Finmark_recal_a.csv
6 1,/Users/cameron/Projects/GSHHG_Land_Mask/GSHHG_Land_Mask.shp,,,
7 0,,,
8 0,,,
9 0,0.006,,,
10 0,82,83,,
11 1,/Users/cameron/Projects/HYPSO_Captures/frohavet/frohavet_2022-11-17_1021Z/
  frohavet_2022-11-17_1021Z.bip,/Users/cameron/Projects/hypso1-agis-gcps/GCPs/frohavet/
  frohavet_2022-11-17_1021Z-bin3.png.points
12 1,/Users/cameron/Projects/HYPSO_Captures/frohavet/frohavet_2022-11-18_1008Z/
  frohavet_2022-11-18_1008Z.bip,/Users/cameron/Projects/hypso1-agis-gcps/GCPs/frohavet/
  frohavet_2022-11-18_1008Z-bin3.png.points
13 1,/Users/cameron/Projects/HYPSO_Captures/frohavet/frohavet_2022-12-10_1012Z/
  frohavet_2022-12-10_1012Z.bip,/Users/cameron/Projects/hypso1-agis-gcps/GCPs/frohavet/
  frohavet_2022-12-10_1012Z-bin3.png.points
14 1,/Users/cameron/Projects/HYPSO_Captures/frohavet/frohavet_2023-03-16_1044Z/
  frohavet_2023-03-16_1044Z.bip,/Users/cameron/Projects/hypso1-agis-gcps/GCPs/frohavet/
  frohavet_2023-03-16_1044Z-bin3.png.points,1
15 1,/Users/cameron/Projects/HYPSO_Captures/frohavet/frohavet_2023-03-28_1059Z/
  frohavet_2023-03-28_1059Z.bip,/Users/cameron/Projects/hypso1-agis-gcps/GCPs/frohavet/
  frohavet_2023-03-28_1059Z-bin3.png.points,1
16 1,/Users/cameron/Projects/HYPSO_Captures/frohavet/frohavet_2023-03-29_1044Z/
  frohavet_2023-03-29_1044Z.bip,/Users/cameron/Projects/hypso1-agis-gcps/GCPs/frohavet/
  frohavet_2023-03-29_1044Z-bin3.png.points,1

```

Figure 3.16: Screen capture of an input file used to generate a time series dataset of the Frohavet location.

The input file format was based around comma separated values (CSV) text files. Each CSV input file consisted of two major sections: the processing settings and the file processing queue. In the processing settings, information related to the processing options of a time series dataset was stored and formatted to be parsed and interpreted by the Python code. Following the processing settings, was the file processing queue. This section of the input file provided file paths to the hyperspectral data files. The file processing queue consisted of a list of .bip files to be included in the dataset. In addition to the .bip files, a list of corresponding .points files was also present to add ground control point (GCP) information to the data files.

Each line of the input file was preceded with a binary line flag. Binary line flags are used in the input files to enable or disable certain parameters, settings, or data files in the time series processing Python scripts. A binary line flag is a configuration parameter that uses a binary value (0 or 1) to indicate the activation or deactivation of a specific setting or capture. In the input file, the line flags are located at the beginning of each line, where value determines whether the corresponding parameter or capture is included/enabled (1) or excluded/disabled (0) during the dataset processing. For the file processing queue, this allows for customization and control over which captures are to be included in the dataset. Individual captures can be selectively added or remove from the dataset using the binary line flags.

In Appendix A, the layout and structure of the dataset processing input files is described. In each subsection, a different line of the input file is presented and broken down into comma separated

variable fields that contain individual parameters or file paths. Figure 3.16 shows an example of a dataset processing input file.

3.5 Independent Component Analysis

3.5.1 Overview

Independent Component Analysis, or ICA, is a powerful technique used to separate individual sources from mixed data. In the case of hyperspectral data, ICA can extract valuable information such as spectral features which each represent a particular physical property or material in the hyperspectral image. In this work, ICA is implemented with a commonly used algorithm called FastICA [34] which is available as a part of the decomposition module in the SciKit-Learn Python library [36]. FastICA is combined with two additional techniques called Automatic Target Generation Process (ATGP) and Principal Component Analysis (PCA). ATGP and PCA serve as components in a method to preinitialize the un-mixing matrix used by FastICA [37].

In each of the change detection techniques used in this project, the use of ICA consisted of four general steps, listed below:

1. ICA Preprocessing
2. ICA Initialization
3. ICA Data Transformation
4. ICA Postprocessing

In the first step, preprocessing, basic operations were applied to center and whiten the data. Additionally, the hyperspectral data was flattened to two dimensional matrices in preparation for FastICA. The second step varied based on the change detection technique but in general it involved preconditioning and initializing the FastICA using results generated by PCA and ATGP, run on a subset or all of the preprocessed hyperspectral data. During the third step, the initialized FastICA algorithm was applied to the preprocessed data, generating a matrix of independent components or ICs. The fourth step, postprocessing, encompassed any remaining tasks related to the reshaping, reconstruction, and interpretation of the independent components generated by FastICA.

3.5.2 ICA Preprocessing

Before applying ICA to hyperspectral data, preprocessing steps are required to ensure the correct application of ICA and effective source separation. Preprocessing attempts to reduce the impact of factors such as noise, uncalibrated data, atmospheric effects, and sensor artifacts. These factors can affect the statistical independence and non-Gaussianity of the components present in the data, of which both properties are fundamental to ICA. Preprocessing techniques like filtering help remove unwanted disturbances and spectral bands while spectral and radiometric calibration ensure consistent measurements across different images.

The most important preprocessing steps for ICA are centering and whitening. Centering enforces that the mean of the ICA input data is equal to zero by subtracting the mean vector from the input

data. Whitening, on the other hand, addresses any preexisting correlations in the data by transforming the data to have a covariance matrix equal to the identity matrix. [34]

Some of these preprocessing tasks, such as spectral and radiometric calibration are performed during the generation of time series datasets. Other preprocessing tasks, such as centering and whitening, must be done immediately prior to running ICA on the data. By applying these preprocessing steps prior to ICA, the hyperspectral data can be optimized for a more accurate decomposition into independent components, leading to more useful results.

Calibration

The data was both spectrally and radiometrically calibrated during the generation of the hyperspectral time series datasets. The implementation of the calibration procedure for the hyperspectral data is discussed in Section 3.4.3.

Filtering

In most hyperspectral captures taken by HYPSON-1, the first four spectral bands consist of all zero values. Because these bands contain little useful information and consume memory and processing resources, the bands are filtered from the data before running ICA. This is done by simply discarding the first four band indices of the NumPy matrix containing the data. After removing the four spectral bands, the hyperspectral data was reduced from a total of 120 bands to 116 bands.

Flattening

Data flattening is the process of converting multidimensional data into a two dimensional matrix, where each row represents a separate sample. This effectively converts each data sample or image pixel into a vector consisting of a series of features or spectral values. Data flattening is performed on the input data since ICA expects the data to be in a two dimensional format where the number of rows corresponds to the number of samples and the number of columns corresponds to the number of features in the input data.

In Python, functions from the NumPy library were used to reshape the data. The `np.reshape` NumPy function was primarily used to convert the hyperspectral data from a three dimensional representation to a two dimensional representation suitable for ICA. Similarly, the same flattening procedure was done on the land mask and the flattened mask was applied to the input data to hide non-water samples. In order to reconstruct the data after running ICA, the original dimensions of the input data were recorded and saved to variables.

Centering

The data centering procedure follows the filtering of spectral bands. Data centering modifies the mean of each band in the hyperspectral data such that it is equal to zero [34]. The motivation of centering is that when the mixture data, \mathbf{x} , has zero mean, then the recovered components, \mathbf{s} , will also have zero mean. Data centering is achieved by calculating a vector containing the means for each spectral band and subtracting it from the data. Because of its criticality to the ICA algorithm, centering is performed automatically the FastICA algorithm as part of the SciKit-Learn signal decomposition module [36].

Whitening

Before running ICA and decomposing the hyperspectral data into components, it is desirable to eliminate any preexisting correlations in the data. By decorrelating the data, the number of parameters in the mixing matrix that need to be estimated can be reduced, in turn reducing the amount of processing required by ICA. A type of linear transformation called a whitening transform is used to decorrelate the data after the data centering process. The whitening transform has several effects on the hyperspectral mixture data. Firstly, it modifies the mixture data \mathbf{x} such that the covariance matrix of \mathbf{x} will be equal to the identity matrix as shown in Equation 2.7. Additionally, the transform also causes the mixture data components to each have variances equal to unity. This is done by scaling the data so that each component has unit variance. After data whitening, the components or spectral bands of \mathbf{x} become statistically uncorrelated from one another. [34]

In the implementation of the data whitening step, a common data whitening method called eigenvalue decomposition (EVD) was used. Similar to data centering, EVD is also built into FastICA algorithm as part of the SciKit-Learn signal decomposition module [36]. The EVD-based data whitening process was automatically invoked and applied to the data whenever the FastICA algorithm was run.

3.5.3 ICA Initialization

One of the features of the SciKit-Learn FastICA algorithm is the optional ability to customize the values used for the initial un-mixing matrix [36]. This initialization ability allows the behavior of FastICA to be modified to generate independent components in a predictable and repeatable manner. By default, the FastICA un-mixing matrix uses values generated from a normal distribution which produces independent components in a random order.

Previous work on ICA and ICA initialization methods has shown that a combination of Principal Component Analysis (PCA) and an algorithm called Automatic Target Generation Procedure (ATGP) can be used to pre-initialize the FastICA algorithm [3]. This initialization strategy involving PCA and ATGP is based around the concept of dimensionality reduction. PCA is used to reduce the dimensionality and simplify the representation of the ICA input data through linearly transforming the data to a new coordinate system. The simplified PCA-transformed data is then used by ATGP to generate initial un-mixing matrix values through an iterative algorithm called Orthogonal Subspace Projection (OSP) [37] [38]. At the end of this process, the un-mixing matrix values computed are used to initialize FastICA before it is run on the ICA input data. The theory and operation of PCA and ATGP are discussed in more detail in Section 2.6.

The initialization technique involving PCA and ATGP addresses several issues with ICA. Firstly, it reduces the amount of computation used to find independent components contributing little to the overall hyperspectral mixture data. PCA and ATGP allow independent components to be generated in order of their significance or contribution to the mixture data. Additionally, since the initial un-mixing matrix values are not drawn from a normal distribution at random, the independent components are generated by ICA in a consistent order each time ICA is run on the same set of data.

PCA

The PCA algorithm was applied to the ICA input data using the SciKit-Learn signal decomposition module [36]. Like for the FastICA algorithm, SciKit-Learn offers built-in native support for PCA. The

only parameter that was required to be changed by PCA was the number of PCA components to be output. This was set to the same number of desired ICA components. For example, if five components were to be generated using ICA, then five components would first be generated with PCA. After running PCA on the ICA input data, a NumPy matrix containing components was created and used as the input to ATGP.

ATGP

The ATGP algorithm was implemented from scratch in Python since there are no pre-existing Python libraries that provide the ATGP algorithm. The ATGP algorithm Python code was partially reused from previous work on ICA initialization techniques [3] and was integrated as a function into the custom `hsts` Python module used for hyperspectral time series dataset processing. The ATGP function iteratively generated the un-mixing matrix \mathbf{W} for the data. This iterative process is the part of ATGP which is based on Orthogonal Subspace Projection (OSP). The NumPy matrix created by PCA in SciKit-Learn and containing the PCA components was used as the source of the initial projection vectors for ATGP and OSP. After running the ATGP algorithm, the output was a matrix \mathbf{U}_n that was then be used as the initial un-mixing matrix \mathbf{W} for FastICA [38].

3.5.4 ICA Data Transformation

For the change detection techniques, ICA was implemented using SciKit-Learn Python library and decomposition module [36] and is based on the work by A. Hyvärinen on the FastICA algorithm [34]. The SciKit-Learn FastICA implementation implements and incorporates some of the ICA preprocessing steps such as the the data centering and whitening preprocessing.

Arguments

The SciKit-Learn FastICA algorithm was used largely unmodified. Two important parameters that were changed, however, were the initial un-mixing matrix `w_init` and the number of iterations. The initial un-mixing matrix was set using FastICA's `w_init` argument and the matrix found using the ATGP algorithm. Setting the un-mixing matrix preconditioned FastICA to generate independent components in order of their significance. The number of ICA iterations was also modified. By default, SciKit-Learn FastICA uses 200 iterations which can lead to a failure of the algorithm converge in some rare cases. This was instead set to 5000 iterations with the intent to guarantee that the FastICA algorithm always converges and to reduce the amount of manual monitoring to make sure that FastICA is converging over the course of multiple runs.

Number of Components

Prior to running ICA, the desired number of independent components needed to be configured. This quantity was set to 10 independent components, the same number of PCA components generated during the ICA initialization process with ATGP. The number of ICA and PCA components must always match for this since it dictates the dimensions of the initial un-mixing matrix for FastICA. The choice of 10 components was arbitrary. In previous work with ICA [3], five components were generated for each HYPSON-1 capture. The reasoning to generate more independent components in this project

was driven by an attempt to find more anomalies within the hyperspectral images as well as to make it easier to match independent components representing the same types of spectral features between temporally separate captures.

Objective Function

The SciKit-Learn FastICA algorithm was also run using the `logcosh` objective function option. This is the default objective function used by SciKit-Learn FastICA but is nonetheless important to mention. The equation is shown in Equation 3.1, where the coefficient a is equal to 1.0. The equation was used to calculate the approximation of negentropy using Equation 2.8. This objective function is one of the proposed equations in the original FastICA paper by A. Hyvärinen [34].

$$G_1(\mathbf{y}) = \frac{1}{a} \log \cosh(a\mathbf{y}) \quad (3.1)$$

Running ICA

The FastICA algorithm was applied to the preprocessed ICA input data using the FastICA transformer's `fit` function. Depending on the specific change detection technique used, ICA was either applied to all of the data all at once or to a subset of the data, such as each capture individually. The application of FastICA produced ICA transformer Python objects containing values and information that were later used to transform and recover the independent components from the data. These tasks were handled and performed in the ICA postprocessing steps.

3.5.5 ICA Postprocessing

Output

The FastICA algorithm outputs two transformer Python objects, one from the FastICA algorithm and a second from the PCA algorithm. These transformer objects each contain several functions, attributes, and variables produced by the ICA and PCA algorithms. The ICA transformer object contains two matrices with the names `components_`, `mixing_`, and `whitening_`. These matrices are the un-mixing, mixing, and whitening matrices, respectively. The matrix `components_` is equivalent to the dot product of the un-mixing matrix and the whitening matrix. The un-mixing matrix is represented by \mathbf{W} in Equation 2.3. The `components_` matrix is in turn equal to the pseudo-inverse of the mixing matrix which is contained in the `mixing_` variable. The PCA transformer has an analogous `components_` matrix which contains the set of PCA components generated by the PCA and ATGP initialization algorithm. The PCA transformer, however, is not necessary for recovering independent components. [36] [87]

The ICA transformer object provides information about the mixing and un-mixing matrices but does not produce the independent components from ICA directly. To account for this, the independent components were computed by transforming the ICA input data using the ICA transformer object's built-in `fit_transform` function. This function applied the ICA transformation previously fit to the data with the ICA transformer's `fit` function. The result of this process was a NumPy matrix containing the independent components. The NumPy matrix had dimensions equal to number of samples (or unmasked pixels in the image) by the number of independent components. In contrast,

the original input data had dimensions equal to the number of samples by the number of features (spectral bands). Because the independent components were flattened and combined into a single NumPy matrix, further processing was required to prepare the independent components to be analyzed and plotted as two dimensional images. [36]

Reshaping

Because the ICA input data was flattening to two dimensions, it had to be reshaped back into multidimensional after running ICA. This process was done simply by reversing the steps taken in the data flattening process; the original dimensions of the input data were read from variables containing them that were created when the data flattening was performed and used to reorganize the data. In Python, functions from the NumPy library were used to reshape the data. The reshape NumPy function was used to convert the hyperspectral data from the flattened two dimensional representation back to a three dimensional representation.

Plotting

Plotting is handled with the Matplotlib Python library. Each independent component was plotted using the ICA transformed data returned as a NumPy array. The spectra associated with each independent component was plotted using the un-mixing matrix `components_` also returned as part of the ICA transformer object. Each row of the un-mixing matrix corresponds to the spectrum of a different independent component.

3.6 Change Detection

3.6.1 Overview

In this project, a variety of ICA-based change detection techniques based on those identified during in the literature review were applied to bitemporal datasets of hyperspectral data. A total five different change detection techniques were implemented in this project. Change detection strategies were varied in each technique used. Owing to their basis on ICA, the selected change detection techniques could primarily be classified as transformation and featured-based change detection techniques. In each techniques, ICA was used at some stage in the processing to extract features from the hyperspectral data. The data was transformed from the image domain to a feature domain that was used to describe or analysis types of changes in the dataset. Algebraic techniques, such as differencing and data stacking, were used too as a component in some of the methods. In these cases, the ICA-based change detection techniques could also be categorized as hybrid change detection approaches.

3.6.2 Relation to ICA

At a high level, two general approaches to ICA-based change detection in hyperspectral imagery were taken. The two general approaches were distinct from one another based on how ICA was applied to the data. The first approach used ICA to find changes directly in the preprocessed bitemporal data. This involved first manipulating the preprocessed bitemporal data in some manner such as through multiple datacube stacking or differencing, where values from an earlier capture are subtracted from

values from a later capture. Typically, these data manipulation actions had the effect of combining or compressing multiple hyperspectral captures to be processed by ICA simultaneously. After performing data manipulation, ICA was run on the modified data. The independent components produced by this process were then each be interpreted as indications of different types of changes or change features. Essentially, independent components in this approach represented non-stationary or changing features across one or more captures. Under this approach, the independent components served directly as change maps describing the spatial distribution.

In comparison, the second approach ran ICA directly on the preprocessed bitemporal data without first modifying it. In this approach, ICA was applied to each hyperspectral capture individually, producing sets of independent components that were temporally distinct from each other. Each set of independent components represented a collection of spectral features from a different temporal moments. Independent components in this approach represented stationary features within a single capture. To produce a change maps, corresponding independent components from different temporal moments were matched with each other using a statistical metric and data differencing was applied.

For the change detection techniques, the ICA processing elements were carried out using the off-the-shelf implementation of FastICA from SciKit-Learn [36]. FastICA was enhanced with an initialization procedure based using Automatic Target Generation Process (ATGP) and Principal Component Analysis (PCA) [3]. All of these components were integrated with various change detection strategies. The implementation of ICA is discussed in further detail in Section XX.

3.6.3 Properties

Multiple considerations were taken into account when selecting ICA-based change detection techniques. These factors aimed to incorporate advantageous and valuable properties that could enhance the interpretability and efficiency of change identification.

Spatial Distribution

To start with, all of the change detection techniques needed to be capable of mapping the spatial distribution of changes. This information helps in identifying regions where a change is occurring and provides spatial context such as the geography of the area. Furthermore, spatial distribution mapping also reveals the extent and pattern of changes in the observed area and enables integration with other localized data sources such as in-situ measurements or satellite observations. In the change detection techniques used for this project, spatial distribution information was provided by change maps derived from independent components. These change maps described the magnitude and direction of changes in multitemporal hyperspectral datasets.

Multiclass Detection

Another desirable property is the ability to distinguish between different types of change. In other words, multiclass change detection techniques are sought after. Hyperspectral imaging can reveal multiple underlying features in a capture so it becomes advantageous to be able to separate and track different kinds of changes individually. When monitoring for HABs, for example, it is favorable to look at algae concentration while disregarding other features that may be present such as sediments,

clouds, or ice. This area is one of the primary strengths of ICA-based change detection since ICA allows hyperspectral data to be decomposed and isolated into signals representing individual features. In summary, ICA augments standard change detection techniques and allows them to be expanded from binary to multiclass change detection techniques.

Interpretable Spectra

Another important consideration is the ability to identify different types of changes. In the context of change detection, it is valuable to not only determine the location and spatial distribution of a change but also to understand the nature and type of the change occurring. This is especially important when dealing with multiclass change detection. ICA fulfills the need for interpretable information about changes by producing independent components that contain spectra information. These spectra are based on information provided by un-mixing matrix (contained in the `components_ICA` transformer variable) and can be analyzed for spectral peaks or troughs reveal the type of material involved in the changing feature. Like its ability to decompose hyperspectral data into individual features, the ability to produce spectral information is another key strength of the ICA-based change detection techniques.

Extendibility

Finally, while not a primary goal of this project, the ability of expanding the change detection techniques from bitemporal to multitemporal change detection was also considered. Not all change detection techniques are appropriate for multitemporal change detection or temporal trajectory analysis, primarily related to how they handle the hyperspectral data. ICA-based change detection techniques that process entire bitemporal time series datasets at once (i.e. through datacube stacking) are less suitable for use on multitemporal datasets. Conversely, techniques that split the time series dataset and process hyperspectral captures individually to retrieve spectral features are easier to efficiently adapt to multitemporal datasets.

3.6.4 Reading Data

Bitemporal pairs of HYPSON-1 hyperspectral captures covering the various observational targets (Section 3.3) were processed by each of the ICA-based change detection techniques separately. The bitemporal datasets consisting of exactly two captures each and always consisted of a before and after capture, taken at two different times. The data was prepared according to the time series dataset processing pipeline described in Section 3.4.

The hyperspectral data was read from NetCDF files using the `xarray` Python library [55]. Each NetCDF file contained exactly one hyperspectral captures and were generated by the dataset processing pipeline described in Section 3.4. Hyperspectral captures of the same observational target were written and saved together into four dimensional NumPy matrices. The four dimensional matrices could be indexed by date in order to load bitemporal capture pairs for use in the ICA-based change detection techniques. The first dimension in the four dimensional matrices represented the time of hyperspectral image acquisition, with lower indices indicating the oldest captures and higher indices indicating the newest captures. The second, third, and fourth dimensions represented height, width, and number of bands of the captures, respectively. To add include land mask information, the four

dimensional matrices were converted to MaskedArray objects using NumPy's masked array module `numpy.ma` [57] with the land mask applied to the data.

Latitude and longitude variables were separated from the data and added to auxiliary NumPy matrices. The geospatial information contained in these matrices were not necessary for the purposes of change detection or ICA but were later used for plotting the independent components and change maps generated by the change detection algorithms. Another auxiliary vector was generated to hold the physical wavelength values of the spectral bands. These labels were calculated from information provided by the spectral calibration files and Python code and were loaded from metadata saved to the NetCDF. The wavelength labels were used for plotting and analyzing reflectance spectra loaded from the ICA un-mixing matrices.

3.6.5 Change Detection Techniques

The five different ICA-based change detection techniques developed for this project were:

1. Spatial Stacking
2. Spectral Stacking
3. Spectral Differencing
4. Hybrid Spectral Differencing and Stacking
5. Component Matching

All of the techniques are based around ICA with ATGP initialization in some form. This means that the five techniques can be categorized as transformation-based or hybrid-based change detection. The techniques were selected according to the properties discussed in the previous subsection. The following sections describe the function and implementation of the ICA-based change detection techniques. In each technique, ICA produced spatial maps of independent components as well as un-mixing matrices containing spectra values. These were interpreted differently depending on the specific ICA-based change detection technique used.

3.6.6 Spatial Stacking

Description

Spatial stacking was the first and simplest ICA-based change detection technique used in this work. In its most basic form, spatial stacking of two hyperspectral datacubes involves combining the pixels from each temporally separate datacube into a single composite datacube. This results in a new datacube with an increased spatial dimension, representing the combined spectral information of both datacubes. In this case, the hyperspectral datacubes were stacked along a mutual vertical axis. Figure 3.17 shows how a pair of before and after datacubes are spatially stacked to form a composite datacube.

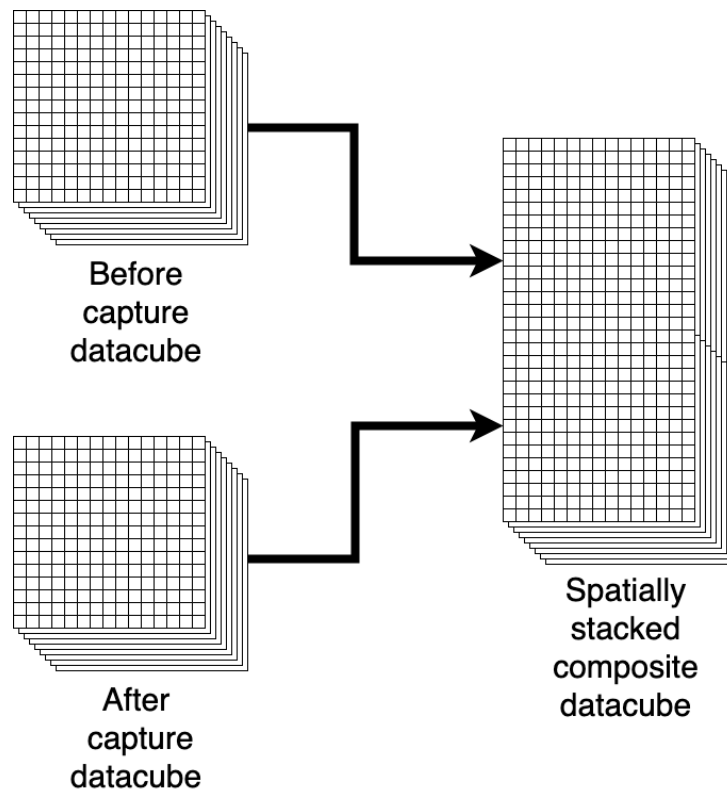


Figure 3.17: Spatial stacking technique.

Once the spatial stacking is performed, ICA can be applied to the composite datacube. By running ICA on the stacked hyperspectral datacube, it becomes possible to identify and extract the independent components, enabling further analysis and interpretation of the combined dataset. The purpose of spatial stacking is so that ICA will identify and generate independent components present in both the before and after hyperspectral datacubes. In this sense, spatial stacking allows ICA to look for features in common between two captures and entire bitemporal dataset.

After producing the independent components for the entire dataset, each independent component can be reshaped and split into two feature maps corresponding to the feature in the before and after capture. This process is effectively a destacking of the independent components along the spatial dimension originally used for the the stacking of the bitemporal datacubes and is shown in Figure 3.18. A simple differencing operation can be applied to each set of before and after feature maps, shown in Figure 3.19. The final result are change maps for each feature identified by ICA. The unmixing matrix from ICA is used to describe the spectra of the independent components.

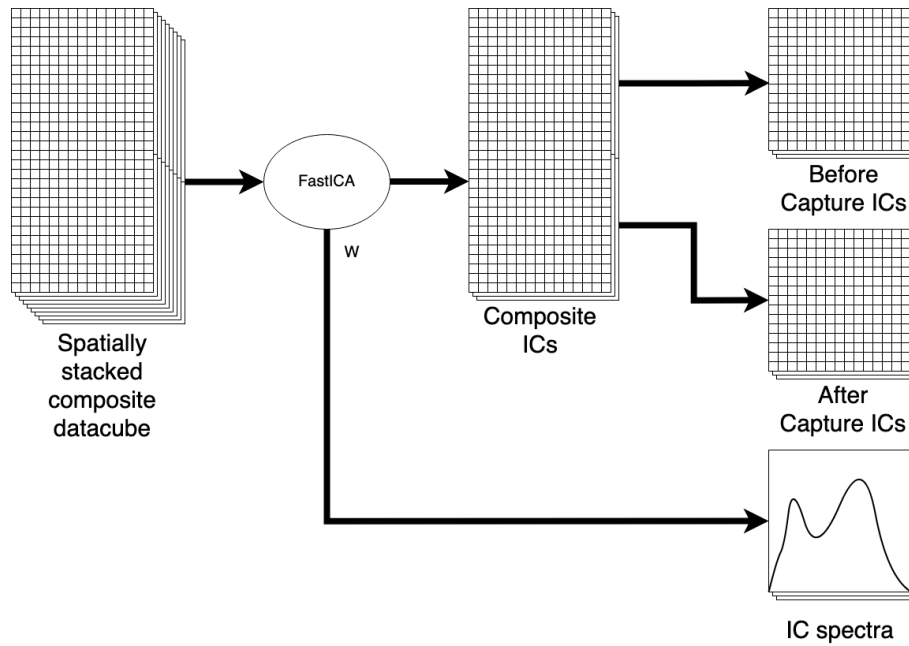


Figure 3.18: Flowchart for the processing of independent components (ICs) and spectra after ICA is run on spatially stacked data.

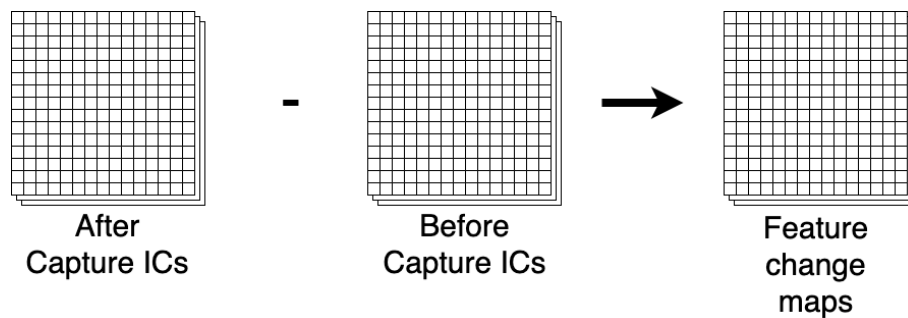


Figure 3.19: Differencing operation on two sets of independent components (ICs) to produce a set of change maps.

Implementation

The implementation of spatial stacking in Python was relatively straightforward. First, a pair of three dimensional hyperspectral datacubes were written to a single four dimensional NumPy masked array with the land mask applied to the data. Next, the two hyperspectral datacubes were reshaped using the NumPy reshape function. The reshaping process converted the bitemporal datacube matrix from dimensions of (2, 522, 308, 116) to dimensions of (1044, 308, 116), changing the vertical dimension from 522 rows to 1044. The non-water pixels were filtered from the reshaped data using NumPy's masked array module `numpy.ma` [57]. The remaining water pixels were copied to a separate ICA input matrix which was subsequently reshaped to two dimensions using the NumPy reshape function. The second reshaping converted the matrix to have dimensions equal to the number of water pixels (of which the quantity was dependent on the scene and land mask) by the number of spectral bands (116). The two dimensional input matrix was then processed with ATGP and ICA.

After processing the data with ICA, another two dimensional output matrix was produced. This output matrix had dimensions equal to the number of water pixels by the number of independent

components (10). Using the land mask and bitemporal datacube matrix, the ICA output data was reconstructed to the same dimensions of the hyperspectral captures, again using the NumPy reshape function. This was effectively the reverse of the procedure used to create the two dimensional ICA input matrix. The data reconstruction produced a new matrix of independent components with dimensions of (2, 522, 308, 10) where the first dimension indicated the capture, the second and third dimensions indicated the height and width of the capture, and the fourth dimension indicated the ICA independent component.

Change maps were generated by applying a simple differencing operations to corresponding independent components, depicted in Figure 3.19. The differencing operation was performed by subtracting the independent component from the first capture from the corresponding independent capture in the second capture. The process was repeated for all 10 pairs of independent components and produced 10 different change maps, each for a different feature. In a final step, geospatial information (latitude and longitude) was added to the change maps. The change maps were plotted and saved using Matplotlib and Basemap Python libraries. [68] [69]

The spectra for the independent components were recovered from the ICA transformer's `components_` attribute. The attribute produced a transposed un-mixing matrix with dimensions of (10, 116). Each row in the matrix was interpreted and plotted as the relative reflectance spectrum of the corresponding independent component using Matplotlib [68].

3.6.7 Spectral Stacking

Description

The second ICA-based change detection technique used was spectral stacking. Spectral stacking is closely related to spatial stacking in that it also combines two hyperspectral datacubes for processing through a stacking operation. As its name suggests, spectral stacking joins temporally separate datacubes along the spectral dimension instead of a spatial dimension as is used in spatial stacking. The spectral stacking process results in a new composite datacube with an increased spectral dimension, also representing the combined spectral information of both datacubes. After performing spectral stacking, ICA can be applied to the composite datacube to extract the independent components of the combined dataset. Spectral stacking is depicted in Figure 3.20.

The reasoning behind spectral stacking is that ICA will identify patterns in the expanded spectral information at each pixel. Because the spectral information for each pixel in the composite datacube includes both before and after spectral values, detectable changes will manifest directly as independent components or features produced by ICA. These features capture the underlying spectral characteristics and changes present in the combined spectral data. The spectral stacking approach offers an efficient way to extract change information without explicitly generating change maps through data differencing. In spatial stacking change detection, the independent components obtained from differencing the independent components of the before and after captures are used to produce change maps. However, by adopting spectral stacking, this intermediate step is bypassed and the need for generating change maps through differencing the independent components is eliminated.

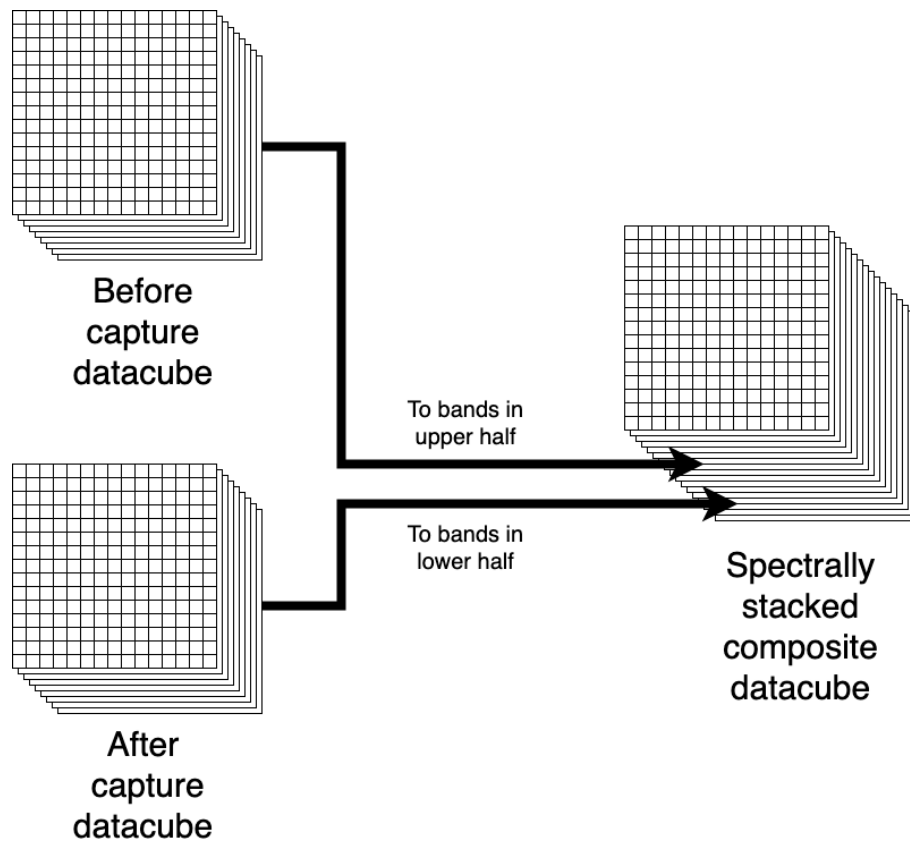


Figure 3.20: Spectral stacking technique.

Following running ICA, very little processing is required to use the independent components. The ICA output simply needs to be reshaped to the original dimensions of the HYPSON-1 captures. Individual independent components can be utilized as change maps describing variations for each feature identified by ICA. The rows in the un-mixing matrix from ICA are divided in half to provide before and after relative reflectance spectra for the hyperspectral captures. The post-ICA manipulation of the independent components and spectra is shown in Figure 3.21.

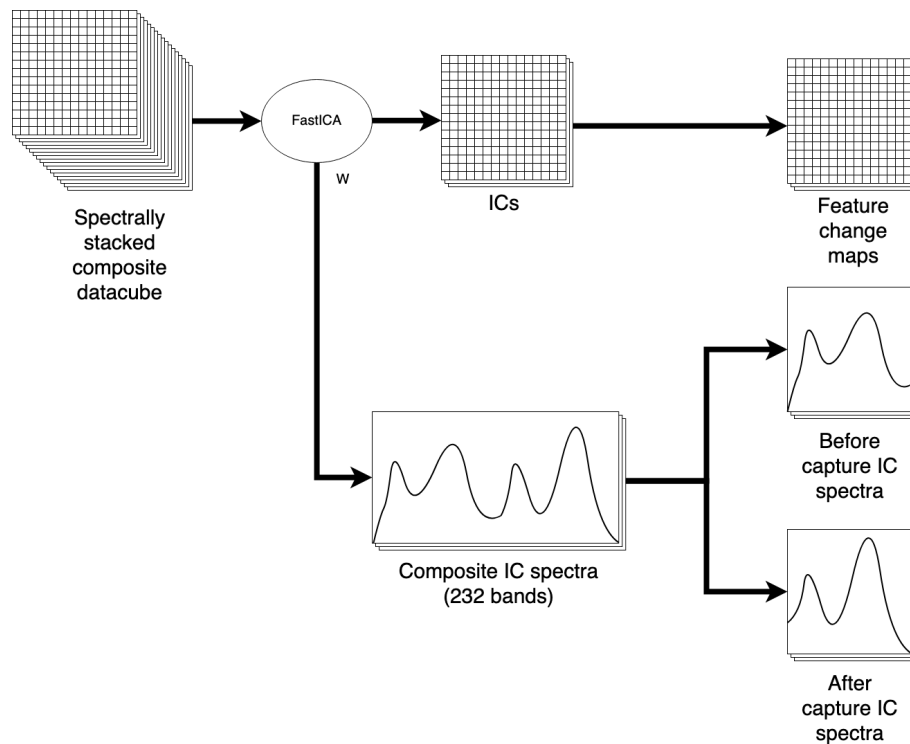


Figure 3.21: Flowchart for the processing of independent components (ICs) and spectra after ICA is run on spectrally stacked data.

Implementation

The implementation of spectral stacking in Python started by writing a pair of three dimensional hyperspectral datacubes to a single four dimensional NumPy masked array with the land mask applied to the data. The hyperspectral datacubes were reshaped using the NumPy reshape function, rearranging the data from bitemporal datacube matrix from dimensions of (2, 522, 308, 116) to dimensions of (522, 308, 232). Note that the spatial dimensions (522 pixels by 308 pixels) of the hyperspectral captures remained unchanged while the spectral dimension doubled from 116 bands to 232 bands.

The next step applied the land mask to the data and filtered non-water pixels from the reshaped data. Again this was done using NumPy's masked array module `numpy.ma` [57]. The filtered data was copied to an ICA input matrix and was flattened to two dimensions with the NumPy reshape function. The ICA input matrix dimensions were equal to the number of water pixels by the number of spectral bands from the two captures (232). The input matrix was then processed with ATGP and ICA.

A two dimensional output matrix containing independent components was produced by ICA. The matrix dimensions were equal to the number of water pixels by the number of independent components (10). The ICA output data was reconstructed to the same dimensions of the hyperspectral captures, again using the NumPy reshape function. This reversed the procedure used to create the two dimensional ICA input matrix. The data reconstruction produced a new matrix with dimensions of (522, 308, 10) consisting of a single set of independent components. Within the resultant matrix, the height and width of the components were indicated by the first and second dimensions and ICA independent components were indicated by the third dimension. Each ICA independent component

in the matrix was interpreted as a different change map. The change maps were plotted and saved using Matplotlib and Basemap Python libraries. [68] [69]

The spectra for the independent components were recovered from the ICA transformer's `components_` attribute. The attribute produced a transposed un-mixing matrix with dimensions of (10, 232). Each row in the matrix was interpreted as relative reflectance spectra values for the corresponding independent component. Because each row in the `components_` matrix had with twice the number of spectral bands than standard HYPSON-1 captures (232 bands instead of 116 bands), they were each divided into two smaller 116 band spectra. These 116 band spectra represented the relative reflectance spectra of the before and after hyperspectral captures in the bitemporal dataset. The pairs of relative reflectance spectra for each independent component were plotted using Matplotlib [68]. Analysis of the before and after relative reflectance spectra could be utilized to reveal what and how specific spectral values changed between the bitemporal captures.

3.6.8 Spectral Differencing

Description

Spectral differencing was the third ICA-based change detection technique. Spectral differencing technique is based on directly analyzing the differences in spectral information between two hyperspectral captures using ICA. The rationale behind spectral differencing is that a differencing operation can be applied to two captures to calculate the amount that each spectral band changed between acquisitions and ICA can be used to reveal distinct types of changes present in the data. Rather than joining two hyperspectral captures using a stacking operation along a mutual spatial or spectral dimension, spectral differencing operates by subtracting corresponding spectral bands of the two datacubes. By subtracting the values of the spectral bands at each pixel location in the captures, a new differential datacube is created, consisting of difference values that highlight the changes in spectral signatures between the two captures.

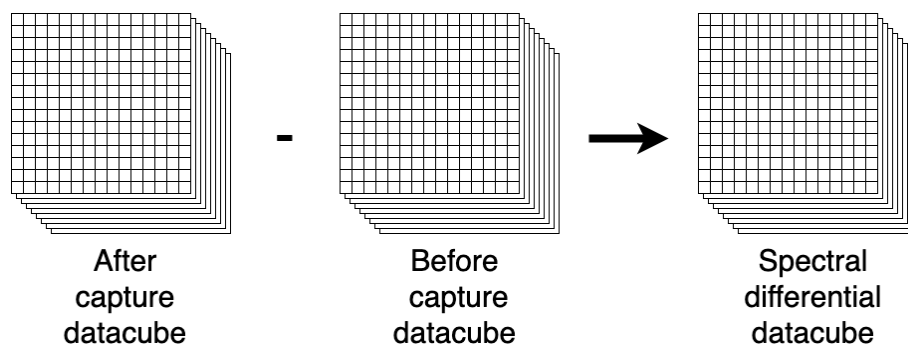


Figure 3.22: Spectral differencing technique.

To further the analysis of changes in the data, ICA is used to identify patterns within the differential datacube. The differential datacube is processed in the same manner as normal hyperspectral data using ICA. In essence, the differential datacube is treated as mixed signal data and undergoes the same ICA processing steps as any other hyperspectral capture.

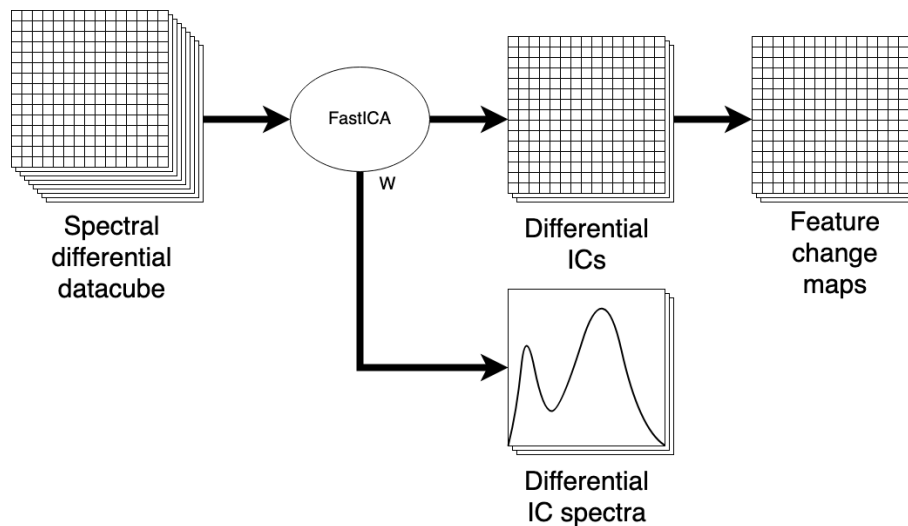


Figure 3.23: Flowchart for the processing of independent components (ICs) and spectra after ICA is run on a differential datacube

Spectral differencing demands minimal processing to use the independent components after running ICA, as seen in Figure 3.23. The ICA output only needs to be reshaped to the original dimensions of the HYPSON-1 captures. The individual independent components produced by ICA can be utilized as change maps, each for a different category of change detected by ICA. In spectral differencing, the un-mixing matrix produced by ICA does not represent the relative reflectance spectra values for static features in the hyperspectral data as it does in the other ICA-based change detection techniques. Instead, the un-mixing matrix represents the change in the relative reflectance spectra values. This property is the main disadvantage of the spectral differencing approach. The spectra from the spectral differencing un-mixing matrix describe how the components change but they do not necessarily describe static spectral signatures of the changing features. As a result, it is difficult to determine what material or physical process is changing using spectral differencing.

Implementation

The Python implementation of spectral differencing started by writing a set hyperspectral time series captures to a four dimensional NumPy masked array with the land mask applied to the data. The masked NumPy matrix was then indexed to obtain two hyperspectral datacubes, one representing data from the before capture and the other representing data from the after capture. The datacubes were contained in three dimensional NumPy matrices, each with dimensions of (522, 308, 116).

Following the loading of the datacubes, the differencing operation was applied to the before and after datacubes. This entailed the element-wise subtraction of the before capture from the after capture, implemented using standard built-in array subtraction in NumPy. The element-wise subtraction produced a new three dimensional differential datacube matrix with the same dimensions as the operand datacubes used in its calculation.

The final steps before running ICA were applying the land mask to the data, filtered non-water pixels using NumPy's masked array module `numpy.ma` [57], and reshaping and flattening the data to an ICA input matrix using the NumPy `reshape` function. The ICA input matrix dimensions were equal to the number of water pixels by the number of spectral bands (116). The input matrix was then

processed with ATGP and ICA.

Once again, a two dimensional output matrix containing independent components was produced by ICA. The matrix dimensions were equal to the number of water pixels by the number of independent components (10). The ICA output data was reconstructed to the same dimensions of the hyperspectral captures, again using the NumPy reshape function. This reversed the procedure used to create the two dimensional ICA input matrix. The data reconstruction produced a new matrix with dimensions of (522, 308, 10) consisting of a single set of independent components. Within the resultant matrix, the height and width of the components were indicated by the first and second dimensions and ICA independent component were indicated by the third dimension. Each ICA independent component in the matrix was interpreted as a different change map. The change maps were plotted and saved using Matplotlib and Basemap Python libraries. [68] [69]

The spectra for the independent components were recovered from the ICA transformer's `components_` attribute. The attribute produced a transposed matrix with dimensions of (10, 116). Each row in the matrix was interpreted as the change in the relative reflectance spectra values for the corresponding independent component or change map. The differential relative reflectance spectra for each independent component were plotted using Matplotlib [68].

3.6.9 Spectral Differencing and Stacking Hybrid

Description

The fourth ICA-based change detection technique developed was hybrid spectral differencing and stacking. The hybrid spectral differencing and stacking technique was developed motivated by the desire to have a change detection method possessing the strengths of both spectral differencing and spectral stacking. Spectral differencing offers a computationally simple method to find and emphasize spectral differences between two datacubes using basic subtraction operations. At the same time, spectral differencing fails to provide easily interpretable information about the static spectral signatures of features in the hyperspectral captures. In contrast, spectral stacking produces spectral signatures of features within the data since ICA operates directly on non-differential hyperspectral data. By combining the two change detection techniques, independent components are generated that incorporate both static and changing spectral signature information.

Hybrid spectral differencing and stacking functions by first subtracting corresponding spectral bands of two datacubes taken from a bitemporal dataset. Subtracting the values of the spectral bands at each pixel location in the captures creates a new differential datacube, consisting of difference values that highlight the changes in spectral signatures between the two captures. From here, the differential datacube is treated as any typical hyperspectral datacube. The differential datacube is combined with the initial hyperspectral datacube (i.e. the before capture) to form a composite datacube. This is achieved through a stacking operation along the spectral dimension, as shown in Figure 3.24. The spectral stacking process results in a composite datacube with an increased spectral dimension that represents the combined spectral information of the initial and differential datacubes. After performing spectral stacking on the differential and initial datacubes, ICA can be applied to the composite datacube to extract the independent components of the combined hybrid dataset.

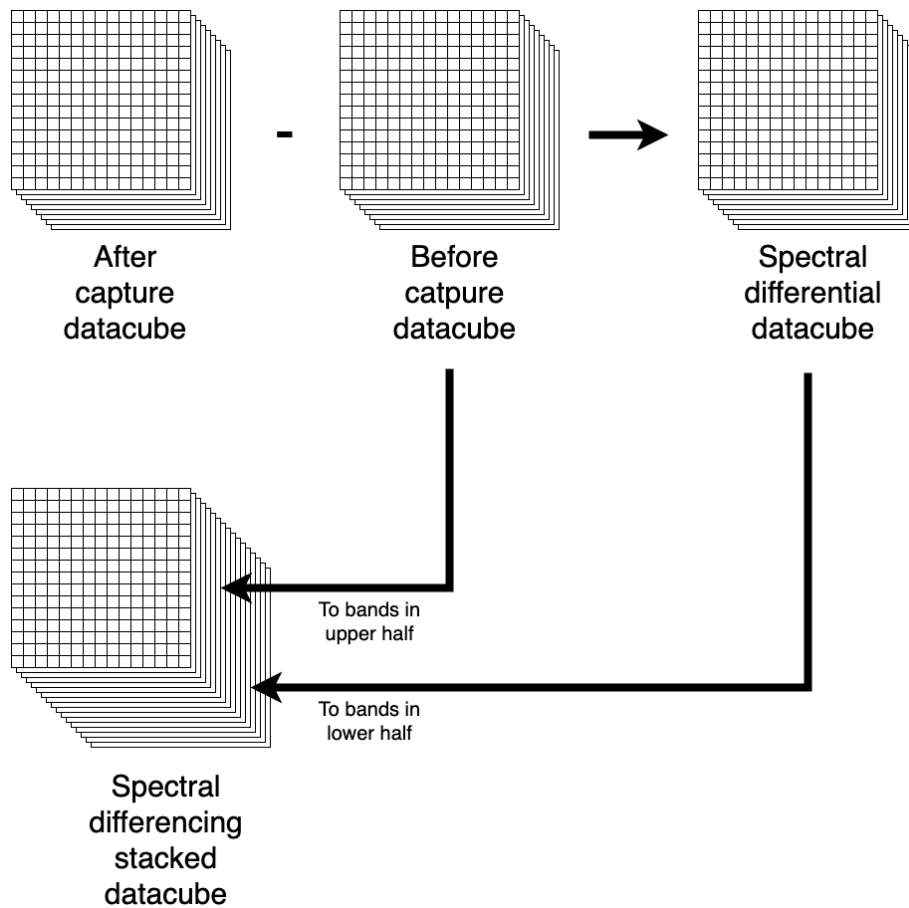


Figure 3.24: Spectral differencing and stacking hybrid technique.

Running on ICA on hybrid spectral differencing and stacking data generates independent components that are used directly as change maps, much as is done in the spectral stacking technique. The independent components only need to be reshaped to the original dimensions of the HYPSON-1 captures. No differencing operation is required after running ICA since it was already performed to create the composite datacube. Within the composite datacube, the spectral information for each pixel includes both the initial values as well as the amount that the spectral values change between the before and after captures. Detectable changes in the present in the composite data will manifest directly as independent components produced by ICA, with each independent component representing a different type of detected change.

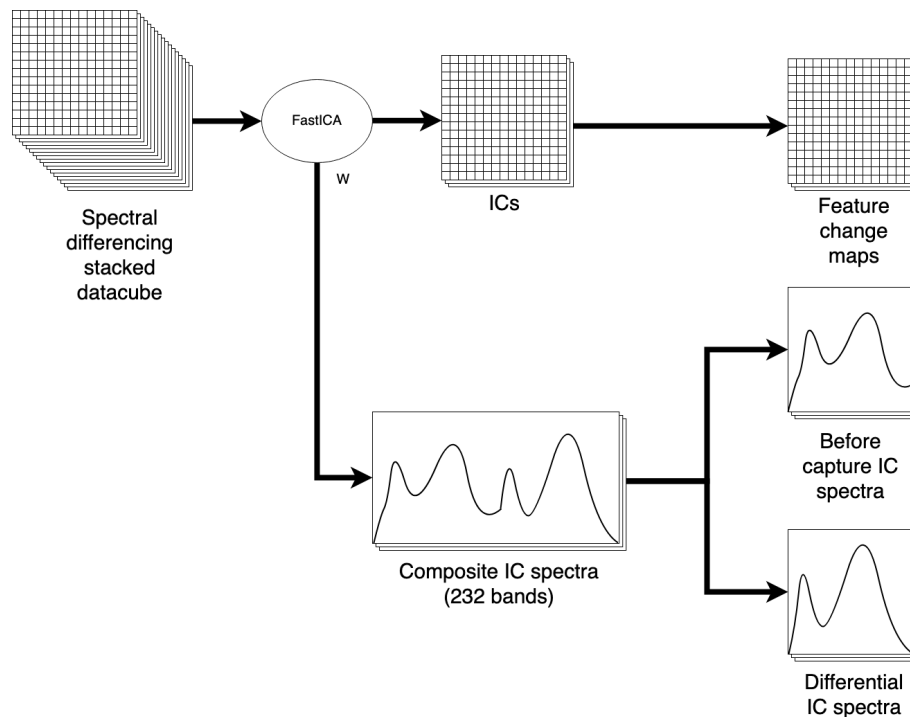


Figure 3.25: Flowchart for the processing of independent components (ICs) and spectra after ICA is run on spectrally differenced and stacked data.

In hybrid spectral differencing and stacking, the spectra contained in the un-mixing matrix have twice the number of spectral bands than a standard HYPSON-1 capture since the hybrid technique partially utilizes spectral stacking. The rows in the un-mixing matrix are divided in half to yield an initial as well as a differential relative reflectance spectrum for each feature in the hyperspectral captures. The initial relative reflectance spectrum can be used to determine what kind of physical material the feature consists of. The differential relative reflectance spectrum describes how the feature changes between the before and after hyperspectral captures. Both spectra assist in the interpretation of the independent components found using the hybrid spectral differencing and stacking technique. Figure 3.25 shows how the independent components and spectra are processed after running ICA.

Implementation

The hybrid spectral differencing and stacking technique was realized in Python. It started by writing a pair of three dimensional hyperspectral capture datacubes to a single four dimensional NumPy masked array with the land mask applied to the data. The four dimensional matrix had dimensions of (2, 522, 308, 116), where the first dimension indicated the index of the bitemporal captures. Next, a differential datacube was first calculated and written to a temporary three dimensional NumPy matrix. The differential datacube was created by element-wise subtracting the values of the first hyperspectral capture datacube from the second hyperspectral capture datacube that were stored in the four dimensional NumPy matrix.

Spectral stacking was then applied to the initial capture datacube and differential datacube. The stacking was done using NumPy operations and created an expanded (522, 308, 232) datacube with 232 spectral bands. This is double the number of bands of a HYPSON-1 capture. The first 116 spectral bands were composed of data from the initial capture datacube. The second 116 spectral

bands were composed of data from the differential datacube.

Immediately following spectral stacking the datacubes, the land mask was applied to the data to filter non-water pixels. This was done using NumPy's masked array module `numpy.ma` [57]. The filtered data was copied to an ICA input matrix and was flattened to two dimensions with the NumPy `reshape` function. The ICA input matrix dimensions were equal to the number of water pixels by the number of spectral bands from the two captures (232). The input matrix was then processed with ATGP and ICA.

ICA produced a two dimensional output matrix containing 10 independent components. The output matrix dimensions were equal to the number of water pixels by the number of independent components (10). The ICA output data was reconstructed to the same dimensions of the hyperspectral captures, again using the NumPy `reshape` function. This reversed the procedure used to create the two dimensional ICA input matrix. The data reconstruction produced a new matrix with dimensions of (522, 308, 10) consisting of a single set of independent components. Within the resultant matrix, the height and width of the components were indicated by the first and second dimensions and ICA independent component were indicated by the third dimension. Each ICA independent component in the matrix was interpreted as a different change map. The change maps were plotted and saved using Matplotlib and Basemap Python libraries. [68] [69]

The un-mixing matrix from the hybrid spectral differencing and stacking technique was loaded from the ICA transformer `components_` attribute. The `components_` attribute produced a transposed un-mixing matrix with dimensions of (10, 232). Each row of the un-mixing matrix corresponded one of the 10 independent components. Because the rows in the un-mixing matrix were 232 elements long, each row in the un-mixing matrix was split into two smaller arrays of 116 elements using NumPy. In each of these rows, the first array of 116 elements was interpreted as the relative reflectance spectrum of the feature in the initial hyperspectral capture. The second array of 116 elements was interpreted as the change in the relative reflectance spectrum (or differential relative reflectance spectrum) of the feature between the time of the initial and final hyperspectral capture. The initial and differential relative reflectance spectra of each independent component were plotted in separately using Matplotlib [68]. Analysis of initial and differential relative reflectance spectra could be utilized to reveal what and how specific spectral values changed between the bitemporal captures.

3.6.10 Component Matching

Description

Component matching is the fifth and final ICA-based change detection technique used in this project. Component matching is the most unique approach of the five techniques since it deviates from the approach to using ICA taken in the first four ICA-based change detection techniques. Instead of first manipulating the hyperspectral data through stacking or differencing operation and then applying ICA, component matching flips the order and applies ICA first to the data and then applies a differencing operation to the resulting independent components. This ordering of processing steps is the second of the two general approaches to ICA-based change detection discussed in Section 3.6.2. Additionally, the component matching technique made several variation to how the ICA algorithm is initialed through the initial un-mixing matrix.

During the application of ICA to the hyperspectral data, the before and after hyperspectral cap-

tures are processed independently by ICA. The use of ICA in this manner produces two sets of independent components, one set for the before capture and one set for the after capture. The goal of component matching is to produce sets of corresponding independent components that represent the same features in the before and after captures, with one component belonging to the before capture and the other belonging to the after capture. Corresponding independent components can then be differenced to produce change maps for each feature picked up on by ICA and present in both the before and after captures.

One of the primary challenges with this approach is the need to find and match the sets of corresponding independent components. Fortunately, the un-mixing matrices produced by the independent executions of ICA on the before and after captures provide easily comparable spectral signatures belonging to the independent components. The matching process of the before capture independent component spectra to the after capture independent component spectra is done using a metric called the Pearson correlation coefficient, or PCC (not to be mistaken with Post-Classification Comparison, also abbreviated by PCC).

The Pearson correlation coefficient is a metric used to score the similarity of two sets of data. It measures the linear relationship between two ordered sequences of data such as arrays. The PCC assigns datasets scores between -1 and 1, with -1 meaning the data is reverse correlated while a score of 1 means the two sequences of data are directly correlated. A score of 0 indicates there is no correlation between the two sequences of data. [88]

The Pearson correlation coefficient $r_{X,Y}$ of two dataset, X and Y , is calculated by dividing the sample covariance of X and Y by the product of the standard deviations of X and Y , as shown in Equation 3.2 [88] [89]:

$$\text{PCC score} = r_{X,Y} = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} \quad (3.2)$$

In this equation, the before and after capture spectra values take the place of the X and Y data sequences.

The Pearson correlation coefficient is well suited for comparison of ICA-generated independent component spectra since the signs (+ or -) of the values in independent components produced by ICA are ambiguous. The sign of any given independent component produced by ICA can either be positive or negative depending on the initial condition of ICA. When running ICA without ATGP initialization, the ICA algorithm can detect and generate the same independent component from the data across multiple consecutive runs, albeit with different signs. Consequently, certain independent components that are produced may have their signs reversed between the repeated runs of ICA [34]. The ambiguous signed nature of independent components means that when comparing independent component spectra from ICA using the PCC, the PCC score's sign can be discarded. Only the magnitude of the PCC score only needs to be considered to determine if the relative reflectance spectra from two different independent components are correlated.

The use of the Pearson correlation coefficient to decide whether two independent components match is straightforward. Larger unsigned PCC scores indicate a closer match between two independent components, suggesting that they belong to the same feature in both the before and after capture. The opposite is true for smaller unsigned PCC scores, where a smaller value indicates and suggests that two independent components do not belong to the same feature.

Since unsigned PCC values can vary from 0 (uncorrelated) to 1 (exact linear correlation), it is use-

ful to utilize threshold or cutoff value to definitively decide if two independent components should be considered a match. A PCC score above the threshold would be classified as a match between the independent components under consideration while a PCC score below the threshold would be classified as a mismatch between the independent components. The threshold value is arbitrary set and is dependent on the capture since the component spectra can vary greatly based on illumination and viewing conditions of the hyperspectral captures. In this project, the cutoff value was usually set between 0.3 and 0.7, selected through trial and error. During trial and error, if too few matches were being found using PCC and the threshold comparison, then the threshold value was lowered.

After finding matches between before and after capture independent components using the Pearson correlation coefficient, change maps for matched features can be produced through simple element-wise differencing. The change maps can be plotted and used to highlight the spatial distribution of the features identified by ICA. Additionally, the spectra associated with the features can be analyzed to identify features based on their spectral signatures. An important consideration is that matched pairs of independent components do not have the exact same spectra since they are produced using data from separate captures and by independent runs of the ICA algorithm. Still, matched independent components should share some similarities such as spectral peaks and troughs at the same spectral bands.

ICA Initialization

One of the unique aspects of component matching is the sequential processing of the hyperspectral captures using ICA. Instead of processing all of the captures at once by a single invocation of ICA, the captures are processed by ICA one by one. This starts with the first capture (before capture), followed by the second (after capture). The sequential processing opens up opportunities to change how each instance of ICA is configured and initialized. In this project, the initialization of the un-mixing matrix was set by the `w_init` argument in the SciKit-Learn FastICA module [36]. This included three different initialization approaches, with each employing a distinct method to initialize ICA for running on the second capture. The first capture was always processed using ICA initialized using PCA and ATGP. The three initialization approaches are described below:

1. **Random un-mixing matrix:** The first ICA initialization approach used the default behavior of FastICA and allowed the initial un-mixing matrix values to be set using random values sourced from a normal distribution [36]. This approach served as a baseline for comparison purposes. Only the ICA transformer for the after capture was initialized with random values; the ICA transformer for the before capture was initialized with the usual PCA and ATGP setup.

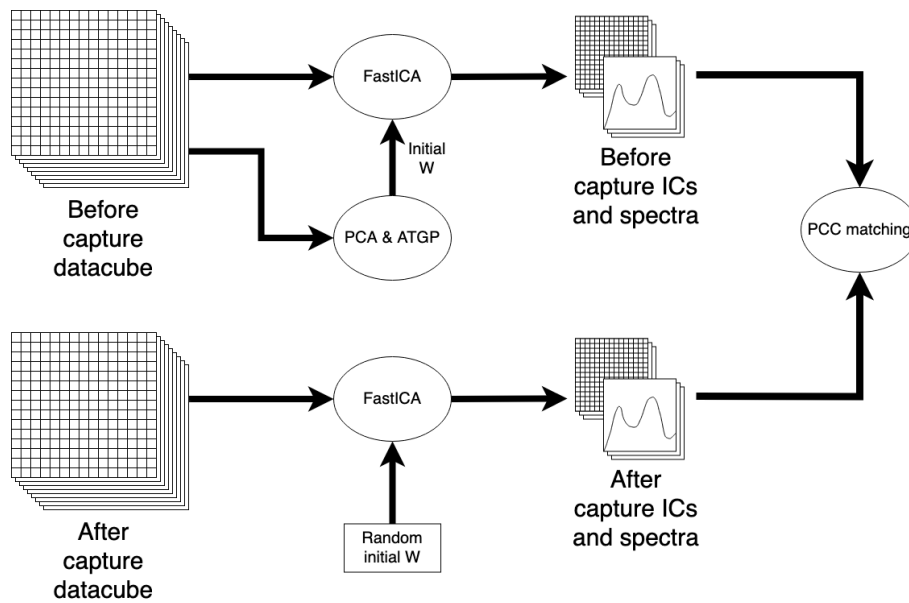


Figure 3.26: Random un-mixing matrix ICA initialization approach.

2. **Previous un-mixing matrix:** The second ICA initialization approach reused the un-mixing matrix from ICA run on the first capture. This approach was chosen since it could reuse information about the independent components present in the first hyperspectral captures to attempt to find the same components in the second capture. In a sense, it allows ICA run on the second capture to pick up where ICA run on the first capture left off and to continue looking for the same components in the second capture.

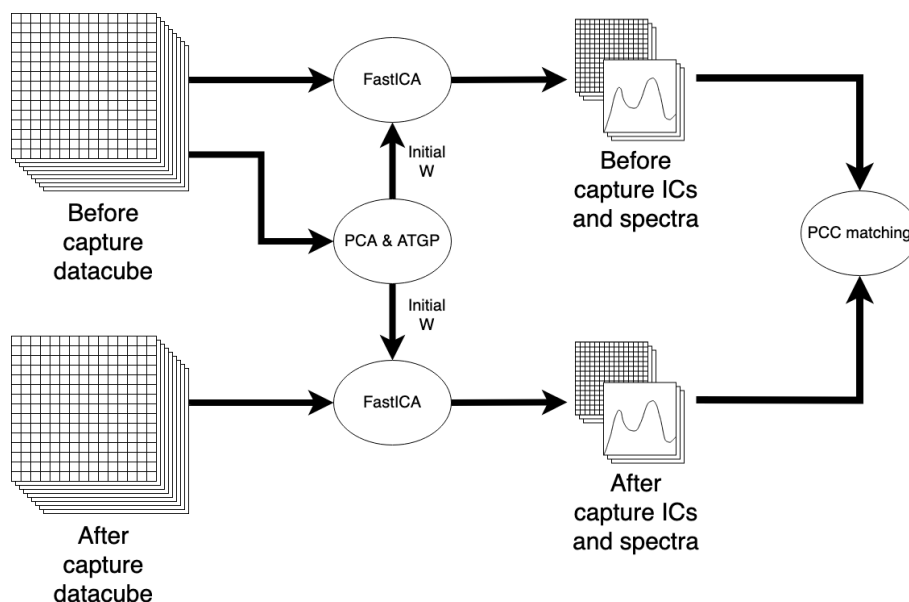


Figure 3.27: Previous un-mixing matrix ICA initialization approach.

3. **ATGP-generated un-mixing matrix:** The third ICA initialization approach was the most complex of the three and utilized the Automatic Target Generation Process (ATGP) algorithm [37]. In this initialization approach, the ICA independent components from the first capture were treated the same as the PCA components generated and used in the standard PCA and ATGP

initialization procedure. The initial ICA un-mixing matrix for the second capture was computed by inputting the independent components from the first capture into the ATGP algorithm. The newly calculated un-mixing matrix was then used to initialize and run ICA on the second hyperspectral capture. Much like the second ICA initialization approach, this approach was chosen to allow information about the independent components present in the first hyperspectral captures to be reused to find the same components in the second hyperspectral capture. It too allows ICA run on the second capture to pick up where ICA run on the first capture left off and to continue looking for the same components in the second capture.

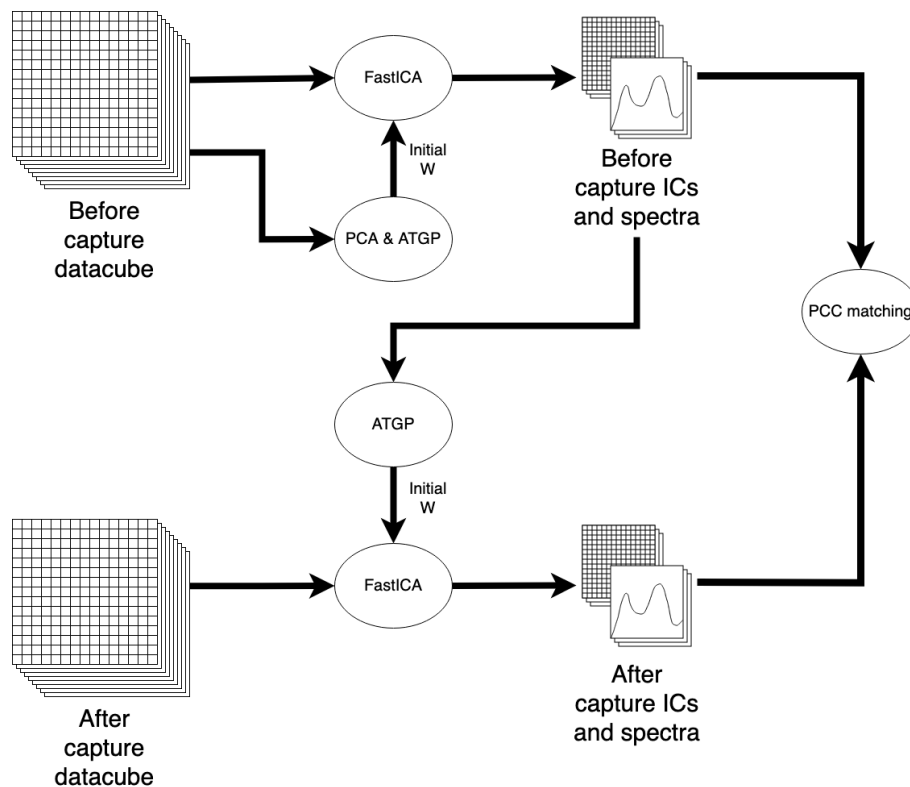


Figure 3.28: ATGP-generated un-mixing matrix ICA initialization approach using previously generated independent components.

Implementation

The implementation of the component matching change detection technique began by writing a set hyperspectral time series captures to a four dimensional NumPy masked array with the land mask applied to the data. The masked NumPy matrix was then indexed to obtain two hyperspectral datacubes, one representing data from the before capture and the other representing data from the after capture. These two before and after datacubes were each written to a separate three dimensional NumPy matrix, each with dimensions of (522, 308, 116).

Next, non-water pixels were filtered from the datacube using NumPy's masked array module `numpy.ma` [57]. The remaining water pixels were copied to a separate ICA input matrices which were subsequently reshaped to two dimensions using the NumPy `reshape` function. The second reshaping converted each input matrix to have dimensions equal to the number of water pixels (of which the quantity was dependent on the scene and land mask) by the number of spectral bands (116).

ICA was run two separate times since there was different ICA input matrix for each capture. The first ICA input matrix processed was the one containing data from the first hyperspectral capture (the before capture). This involved using ATGP and ICA to process the first capture in a process identical to those used in the other ICA-based change detection techniques. First, PCA was run on the capture to produce a set of PCA components. The PCA components were then fed into the ATGP algorithm which produced an initial un-mixing matrix for FastICA. Finally, ICA was run and produced a set of 10 independent components for the first hyperspectral capture.

The 10 independent components from the first capture were contained within a two dimensional output matrix. The output matrix dimensions were equal to the number of water pixels by the number of independent components (10). The ICA output data from the first capture was reconstructed to the same dimensions of the hyperspectral captures, again using the NumPy reshape function. This reversed the procedure used to create the two dimensional ICA input matrices. The data reconstruction produced a new matrix with dimensions of (522, 308, 10) consisting of a single set of independent components. Within the resultant matrix, the height and width of the components were indicated by the first and second dimensions and ICA independent components were indicated by the third dimension. The un-mixing matrix containing the independent component spectra from the first capture was contained in the ICA transformer `components_` attribute. The `components_` attribute produced a transposed un-mixing matrix with dimensions of (10, 116). Each row of the un-mixing matrix corresponded one of the 10 independent components.

Processing the second hyperspectral capture (the after capture) was configured differently from the first hyperspectral capture. The exact procedure use for decomposing the second capture into independent components varied based on the initialization approach for FastICA. These procedures are described individually below:

1. **Random un-mixing matrix:** Random un-mixing matrix initialization took no interventions to initialize the un-mixing matrix in SciKit-Learn FastICA . The `w_init` argument in FastICA was left empty. The empty `w_init` argument had the effect of initializing the un-mixing matrix values using random values sourced from a normal distribution [36].
2. **Previous un-mixing matrix:** Previous un-mixing matrix initialization reused the same un-mixing matrix computed by PCA and ATGP and used for initializing ICA for the first hyperspectral capture. The un-mixing matrix was retrieved from the first ICA transformer using the `w_init` attribute. The NumPy matrix returned by the `w_init` attribute was then used for the `w_init` argument in the ICA transformer for the second capture.
3. **ATGP-generated un-mixing matrix:** ATGP-generated un-mixing matrix initialization was based on a variation of the Automatic Target Generation Process (ATGP) algorithm [37]. FastICA produced a NumPy matrix of independent components from the first capture with dimensions equal to (522, 308, 10). These dimensions were the same as the NumPy matrix produced by PCA when it is used to transform the hyperspectral captures to PCA components. Instead of using PCA components, the NumPy matrix of independent components from the first capture was used as the input for ATGP algorithm, callable using the `hsts.ica.atgp` function. The `hsts.ica.atgp` function only required arguments for the matrix of independent components and number of ICA components (10). PCA was not run on the second hyperspectral capture and was omitted from the Python code. The `hsts.ica.atgp` returned an initial un-mixing

matrix contained in a NumPy matrix. The NumPy matrix used for the `w_init` argument in the ICA transformer for the second capture.

After initializing FastICA, ICA was run and produced a set of 10 independent components for the second hyperspectral capture. Much like the independent components from the first capture, the 10 independent components from the second capture were contained within a two dimensional output matrix. The output matrix dimensions were equal to the number of water pixels by the number of independent components (10). The ICA output data from the second capture was reconstructed to the same dimensions of the hyperspectral captures, again using the NumPy `reshape` function. This reversed the procedure used to create the two dimensional ICA input matrices. The data reconstruction produced a new matrix with dimensions of (522, 308, 10) consisting of a single set of independent components. Within the resultant matrix, the height and width of the components were indicated by the first and second dimensions and ICA independent components were indicated by the third dimension. The un-mixing matrix containing the independent component spectra from the second capture was contained in the ICA transformer `components_` attribute. The `components_` attribute produced a transposed un-mixing matrix with dimensions of (10, 116). Each row of the un-mixing matrix corresponded one of the 10 independent components.

The component matching process was begun after producing sets of independent components for both before and after captures. The un-mixing matrices from both captures were loaded from the ICA transformer objects and `components_` attribute. The rows of the un-mixing matrices were split into individual NumPy arrays, 116 elements in length. Each NumPy array created possessed the spectral values corresponding to one of the independent component. Due to high variability and fluctuations in the spectral bands at lower wavelengths, the first 40 spectral values were omitted from the NumPy arrays. The cause of the fluctuations in the data at lower spectral bands is uncertain but is likely due to sensor artifacts in HYPSON-1. The remaining spectral values contained in the NumPy arrays were the data sequences that were compared using the Pearson correlation coefficient.

Independent component spectral values from the first and second hyperspectral captures were compared with the Pearson correlation coefficient within a Python for-loop. First, each sequence of spectral values was first normalized to values between -1 and 1. Then, each sequence of normalized spectral values from the first capture was compared and scored against every sequence of normalized spectral values from the second capture.

The `scipy.stats.pearsonr` Pearson correlation coefficient module from SciPy was used to calculate the PCC scores. The function `scipy.stats.pearsonr` accepted two arrays of data, x and y , as input and returned a PCC score for the two array with a value between -1 and 1 [88]. The absolute values of the PCC scores were taken in order to remove the sign and convert negative values to positive values. This was done find matches between independent components that only differed by a sign change and were still correlated. The PCC scores and pairs of before and after independent components were recorded to a Python list. The list of PCC scores was reordered from highest PCC scores to lowest PCC scores using a Python lambda function.

A threshold value was applied to the array of scores to cut off independent component pairs with a PCC score below a certain value. This was done using built-in logical operators in Python. The remaining independent component pairs were considered matched pairs of features from the first and second hyperspectral captures. The threshold value usually set between 0.3 and 0.7 and was selected through trial and error. If too few matches were generated by PCC and the threshold, then

the threshold value was changed to a lower value.

After running ICA and matching the independent components from the before and after captures, all that remained was to generate change maps for the various features extracted by ICA. This was done by applying a differencing operation to the matched independent components. The differencing operation was performed on each matched pair of components by subtracting the first independent component in the matched pair from the second independent in the matched pair. The process was repeated for all of the matched pairs of independent components with PCC score above the threshold. Each differencing operation created a change map for a different spectral feature. In a final step, geospatial information (latitude and longitude) was added to the change maps. The change maps were plotted and saved using Matplotlib and Basemap Python libraries. [68] [69]

In addition to the change maps, each feature had a set of two relative reflectance spectra with one taken from the first capture's ICA un-mixing matrix and the other taken from the second capture's ICA un-mixing matrix. The relative reflectance spectra were plotted alongside the feature change maps using Matplotlib [68]. Analysis of before and after relative reflectance spectra could be utilized to reveal what and how specific spectral values changed between the bitemporal captures.

3.7 Validation Data

Validation data plays a crucial role in evaluating and improving the performance of the hyperspectral change detection techniques. The purpose of validation data is to provide reliable reference information about the changes that have occurred between subsequent hyperspectral captures. Validation data serves as a benchmark against which the change detection technique results are compared, allowing for the assessment of their accuracy and effectiveness. The consistency between the features observed in the hyperspectral data and change maps generated by HYPSON-1 and the ICA-based change detection techniques can be confirmed by cross-validating them with features present in validation data acquired at similar times by other imaging satellites.

In this work, a focus was placed on validation data that provided information about chlorophyll and suspended matter concentrations in bodies of water. In particular, chlorophyll pigment concentration can be used to indicate regions in hyperspectral images where a harmful algal bloom may be present [90]. To this end, two data validation sources were utilized, with the first data validation data source being the National Oceanic and Atmospheric Administration's (NOAA) Lake Erie Harmful Algal Bloom Forecast [74] and the other validation data source being the Sentinel 3 Ocean Land Colour Instrument (OLCI) [75].

3.7.1 NOAA HAB Forecast Validation Data

The Lake Erie Harmful Algal Bloom Forecast [74] is one of several algal bloom forecast products developed and produced by the National Oceanic and Atmospheric Administration's (NOAA) National Centers for Coastal Ocean Science (NCCOS) in the United States. It monitors bloom conditions of a type of blue-green algae called cyanobacteria which can cause seasonal HABs in Lake Erie, one of the Great Lakes in the eastern United States. The Lake Erie HAB forecast is a regional product and only has coverage of Lake Erie. Additionally, the Lake Erie HAB forecast is only produced during the northern hemisphere summer when the water in Lake Erie is warm enough for algal bloom to occur. The

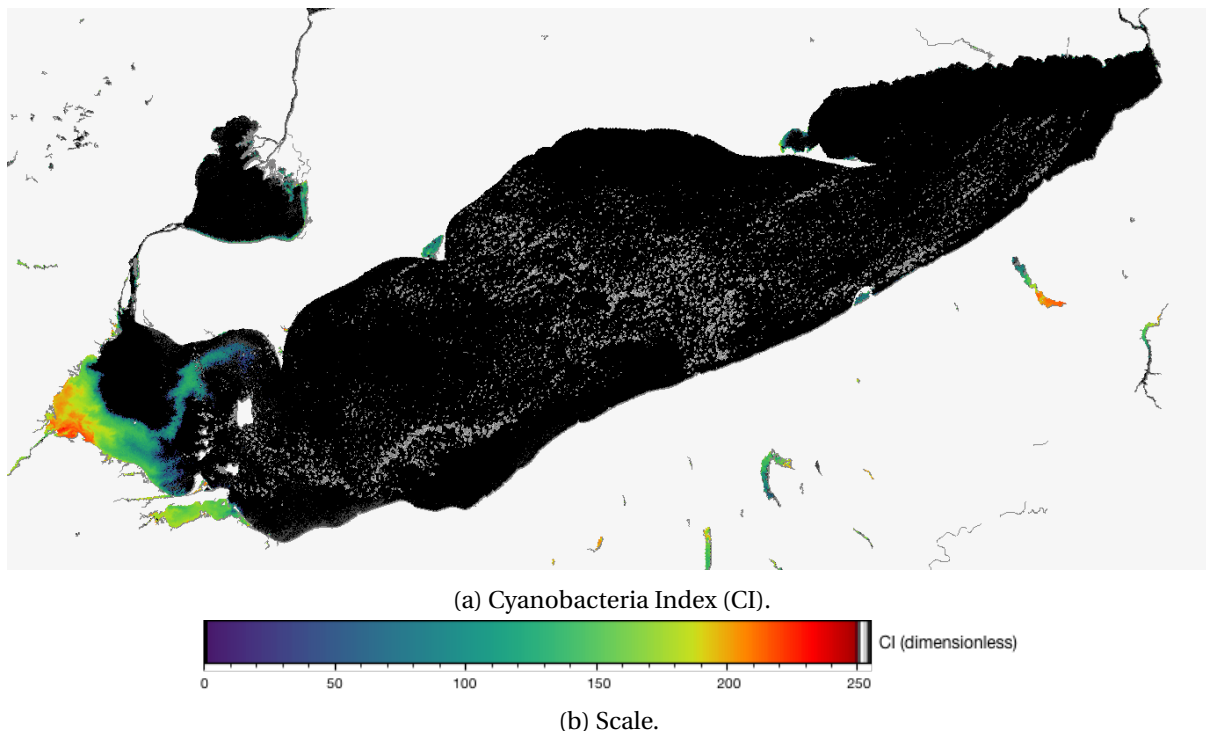


Figure 3.29: Lake Erie NOAA HAB Forecast for 27 August 2022 15:54 UTC

Lake Erie HAB forecast products include both an observed bloom and modelled forecast bloom products. The information used in Lake Erie HAB forecast is a combination of data sources from in-situ monitoring at Lake Erie as well as multispectral satellite observations from the Ocean Colour Land Imager (OLCI) on the European Space Agency's Sentinel-3 Earth observation satellites.

For the purposes of this project, a Lake Erie HAB product measuring a quantity called the Cyanobacteria Index (CI) was used as validation data. The cyanobacteria index measures the relative abundance of cyanobacteria (blue-green algae) biomass present using the formula in Equation 3.3. The variables ρ signify the top-of-atmosphere reflectance values at a certain wavelength:

$$CI = -1 * \left(\rho_{\lambda_2} - \rho_{\lambda_1} + (\rho_{\lambda_3} - \rho_{\lambda_1}) * \frac{\lambda_2 - \lambda_1}{\lambda_3 - \lambda_1} \right) \quad (3.3)$$

The wavelengths used in calculating the index using the formula are $\lambda_1 = 665$ nm, $\lambda_2 = 681$ nm, and $\lambda_3 = 709$ nm [91] [92]. Because the cyanobacteria index measures the relative abundance of cyanobacteria, the index is dimensionless and does not provide an exact quantity for the concentration of cyanobacteria or chlorophyll [92]. Despite this, the cyanobacteria index can still be used to determine the spatial distribution or potential HABs and be used for qualitative comparisons with independent components and change maps generated using HYPSON-1 observations.

The cyanobacteria index products are distributed as georeferenced GeoTIFF image files along with a true color image [92]. Within the GeoTIFF files, pixels such as clouds, land, or other invalid observations are flagged, allowing them to be easily omitted. Values in the cyanobacteria index GeoTIFFs are dimensionless and range from 0 to 250. The GeoTIFF files were downloaded from the NOAA HAB Data Explorer website [93]. NOAA's NCCOS produces one cyanobacteria index GeoTIFF image covering the Lake Erie basin each day. One issue encountered while acquiring the GeoTIFF files, is the availability of historical data. The NOAA HAB Data Explorer site only maintains a

historical archive of GeoTIFFs extending back approximately half a year. To ensure continued access to the files, all of the downloaded GeoTIFF file were saved locally on personal computers. Figure 5.10 shows one of the Lake Erie HAB forecast GeoTIFF images displayed next to the true color image of the same scene.

The cyanobacteria index GeoTIFF files were not loaded into Python as was done for HYPSON-1 and Sentinel-3 data. Instead the GeoTIFFs were opened and viewed using QGIS for qualitative comparison with the independent components and change maps generated using HYPSON-1 observations. QGIS has native support for viewing GeoTIFF as well as the ability to easily hide invalid pixels. Furthermore, GeoTIFFs in QGIS can be overlaid on existing geospatial data for analysis or be exported to other image formats.

3.7.2 Sentinel-3 OLCI Validation Data

The other validation data source was the European Space Agency's Sentinel-3 Earth observation satellites. Sentinel-3 is a satellite mission developed by the European Space Agency (ESA) as part of the Copernicus program. It is designed to monitor Earth's oceans, land surfaces, and atmosphere for environmental and climate-related applications. The Sentinel-3 mission consists of two satellites, Sentinel-3A and Sentinel-3B, which were launched in 2016 and 2018, respectively. Data is processed and delivered from the Sentinel satellites in near-real time, allowing it to be used for operational ocean monitoring purposes. Furthermore, the multispectral imagery data collected by Sentinel-3 is freely available to users and has global coverage. [94]

One of the key instruments aboard Sentinel-3 is the Ocean and Land Colour Instrument (OLCI). The OLCI is a push-broom imaging spectrometer specifically designed to provide multispectral measurements of ocean and land surface color. It operates in the visible and near-infrared spectral ranges, from 0.4 to 1.02 μm , capturing data across 21 spectral bands. This multispectral capability enables the retrieval of essential information and allows the multispectral data to be processed into useful ocean and land measurement data products. [75]

Sentinel-3 has a collection of full resolution ocean products called OL_2_WFR [95]. It consists of 24 separate measurement files that are primarily stored as NetCDF files. Within the collection of Sentinel-3 data products available, three Level-2 data products were directly relevant as validation data for the ICA-based change detection algorithms. These products were chl_oc4me, chl_NN, and TSM_NN. The first two data products, chl_oc4me and chl_NN, provided measurements of chlorophyll-a (Chl-a) algal pigment concentrations in water. The third data product, TSM_NN, provided measurements of total suspended matter in bodies of water, such as sediments or water run-off.

The chlorophyll concentrations in the chl_oc4me and chl_NN data products were calculated using one of two methods. In the first method, used to produce the chl_oc4me data product, a Maximum Band Ratio (MBR) algorithm called OC4Me was used to calculate chlorophyll concentration. The OC4Me algorithm computed algal concentration from a collection of four spectral bands collected by the OLCI:

$$\log_{10} \text{Chl-a} = A_0 + A_1(\log_{10} R) + A_2(\log_{10} R)^2 + A_3(\log_{10} R)^3 + A_4(\log_{10} R)^4 \quad (3.4)$$

The variable R_j^i is the ratio of the reflectance at band R_i over R_j , where j is 560 nm. The band i is

either 443, 490, or 510 nm and is selected to maximize the ratio:

$$R_j^i = \max_{i,j} \left(\frac{R_i}{R_j} \right) \quad (3.5)$$

The remaining coefficients are $A_0 = 0.45027$, $A_1 = -3.259491$, $A_2 = 1.9743$, $A_3 = 3.522731$, and $A_4 = 0.949586$ [96] [25].

The second method of computing algal concentration data products was a neural network based approach called Inverse Modelling Technique (IMT). The IMT neural network used a Inverse Radiative Transfer Model-Neural Network (IRTM-NN) to estimate the absorption coefficient of algal pigment (indicating chlorophyll). The IMT neural network was also used to estimate the absorption coefficient for suspended matter concentration in water. The absorption coefficients were derived into data products for chl_NN and TSM_NN. [97]

Unlike the NOAA Lake Erie HAB data, which uses the Cyanobacteria Index (CI) to describe relative abundance of chlorophyll in water as a dimensionless quantity, the chlorophyll and total suspended matter concentration products from Sentinel-3 provide values with units of mass per volume. For the chlorophyll products in chl_oc4me and chl_NN, chlorophyll concentration is provided with units of mgm^{-3} (Chl-a). Total suspended matter concentration in TSM_NN is provided with units of gm^{-3} .

The ocean data products of Sentinel-3 OLCI OL_2_WFR were obtained from the Onda Dias online data catalog [98]. This data catalog serves as a repository for data archives from various missions, including Sentinel-3 and other Copernicus missions. Sentinel-3 granules can be searched for through the site's graphical browser or by utilizing location, date, and time keywords to refine the search. The Sentinel-3 granules that best matched the time and location of the HYPSON-1 captures used by the change detection algorithms were obtained by downloading the corresponding data from the Onda Dias catalog. The Sentinel-3 OL_2_WFR data was downloaded as .zip files, each generally on the order of several hundred megabytes in size.



```

_0179_069_382_2160_MAR_0_NT_003.SEN3 cameron$ ls -l
total 476896
-rw-r--r--@ 1 cameron staff 7796552 Aug 28 2022 0a01_reflectance.nc
-rw-r--r--@ 1 cameron staff 7706299 Aug 28 2022 0a02_reflectance.nc
-rw-r--r--@ 1 cameron staff 7561143 Aug 28 2022 0a03_reflectance.nc
-rw-r--r--@ 1 cameron staff 7278918 Aug 28 2022 0a04_reflectance.nc
-rw-r--r--@ 1 cameron staff 7199783 Aug 28 2022 0a05_reflectance.nc
-rw-r--r--@ 1 cameron staff 7002917 Aug 28 2022 0a06_reflectance.nc
-rw-r--r--@ 1 cameron staff 6694853 Aug 28 2022 0a07_reflectance.nc
-rw-r--r--@ 1 cameron staff 6421017 Aug 28 2022 0a08_reflectance.nc
-rw-r--r--@ 1 cameron staff 6372999 Aug 28 2022 0a09_reflectance.nc
-rw-r--r--@ 1 cameron staff 6317391 Aug 28 2022 0a10_reflectance.nc
-rw-r--r--@ 1 cameron staff 6173359 Aug 28 2022 0a11_reflectance.nc
-rw-r--r--@ 1 cameron staff 5837664 Aug 28 2022 0a12_reflectance.nc
-rw-r--r--@ 1 cameron staff 5070074 Aug 28 2022 0a16_reflectance.nc
-rw-r--r--@ 1 cameron staff 5020881 Aug 28 2022 0a17_reflectance.nc
-rw-r--r--@ 1 cameron staff 5009402 Aug 28 2022 0a18_reflectance.nc
-rw-r--r--@ 1 cameron staff 7018280 Aug 28 2022 0a21_reflectance.nc
-rw-r--r--@ 1 cameron staff 5260919 Aug 28 2022 chl_nn.nc
-rw-r--r--@ 1 cameron staff 4724769 Aug 28 2022 chl_oc4me.nc
-rw-r--r--@ 1 cameron staff 62169236 Aug 28 2022 geo_coordinates.nc
-rw-r--r--@ 1 cameron staff 1065046 Aug 28 2022 instrument_data.nc
-rw-r--r--@ 1 cameron staff 4866589 Aug 28 2022 iop_nn.nc
-rw-r--r--@ 1 cameron staff 14542591 Aug 28 2022 iwv.nc
-rw-r--r--@ 1 cameron staff 1900251 Aug 28 2022 par.nc
-rw-r--r--@ 1 cameron staff 1232831 Aug 28 2022 tie_geo_coordinates.nc
-rw-r--r--@ 1 cameron staff 2239934 Aug 28 2022 tie_geometries.nc
-rw-r--r--@ 1 cameron staff 19150202 Aug 28 2022 tie_meteo.nc
-rw-r--r--@ 1 cameron staff 15470 Aug 28 2022 time_coordinates.nc
-rw-r--r--@ 1 cameron staff 3426823 Aug 28 2022 trsp.nc
-rw-r--r--@ 1 cameron staff 5234293 Aug 28 2022 tsm_nn.nc
-rw-r--r--@ 1 cameron staff 4825471 Aug 28 2022 w_aer.nc
-rw-r--r--@ 1 cameron staff 8784355 Aug 28 2022 wqsf.nc
-rw-r--r--@ 1 cameron staff 189649 Aug 28 2022 xfdumani_fest.xml
(base) Delphinus:S3B_OL_2_WFR____20220827T155148_20220827T155448_20220829T014235
_0179_069_382_2160_MAR_0_NT_003.SEN3 cameron$ █

```

Figure 3.30: The typical set of data files included in the Sentinel-3 OLCI OL_2_WFR data product.

Within each OL_2_WFR .zip file, there were individual data products and auxiliary files stored as .nc NetCDF files. The file structure of the Sentinel-3 OLCI OL_2_WFR ocean product is shown in Figure 3.30. Some of the NetCDF files were measurement files for individual reflectance bands. Others, like tsm_nn.nc and chl_nn.nc, were processed Level-2 data products derived from the reflectance values. The files also included auxiliary files containing georeferencing and sensor parameters. Latitude and longitude values were stored separately from the reflectance and processed products in a NetCDF file titled geo_coordinates.nc. [99]

The Sentinel-3 OLCI OL_2_WFR products were loaded into Python for plotting and comparison with the HYPSON-1 data. This was done using a Python library called SatPy which maintains numerous file readers for various sensors including for Sentinel-3 OLCI Level-2 data. SatPy was used to load data from chl_nn.nc, tsm_nn.nc, geo_coordinates.nc. The OC4Me chlorophyll concentration product from chl_oc4me.nc was left out because it was not available in all of the scenes used as validation data. A geometry area was defined in SatPy using the latitude and longitude extents of the HYPSON-1 change maps. This allowed the Sentinel-3 data to be plotted and displayed in the same location and extents as the HYPSON-1 data. SatPy automatically handled resampling of the Sentinel-3 data using the pyresample submodule.

The Sentinel-3 OLCI OL_2_WFR products were loaded into Python for plotting and comparison with the HYPSON-1 data. To accomplish this, the SatPy Python library [65], which supports various

file readers for different sensors including Sentinel-3 OLCI Level-2 data, was utilized. The data from `chl_nn.nc`, `tsm_nn.nc`, and `geo_coordinates.nc` were loaded using SatPy. However, the OC4Me chlorophyll concentration product from `chl_oc4me.nc` was excluded from the analysis since it was not available in all of the scenes used as validation data.

To align the Sentinel-3 data with the HYPSON-1 data, a geometry area was defined within SatPy. This geometry area was created using the latitude and longitude extents taken from the geographic bounding boxes of the HYPSON-1 change maps. This approach allowed the Sentinel-3 data to be plotted and displayed within the same location and extents as the HYPSON-1 data. SatPy handled the resampling of the Sentinel-3 data automatically through the `pyresample` submodule, ensuring accurate alignment with the HYPSON-1 data. The resampled and aligned Sentinel-3 Chl-a and TSM data was written to NumPy matrices, accompanied by matrices containing latitude and longitude values for the pixels.

In the NetCDF files, the Sentinel-3 Chl-a and TSM data were initially stored in a logarithmic format, having undergone a logarithmic transformation. To obtain the physical concentration values for chlorophyll and total suspended matter, the logarithmic transformation was reversed for the data. The reversal yielded the original non-logarithmic concentration values of the variables that could be used as validation data.

In order to generate Sentinel-3 change maps for validating the HYPSON-1 change maps, matching pairs of Sentinel-3 Chl-a and TSM NumPy matrices were selected based on their closest match in terms of their acquisition time to the HYPSON-1 captures. The difference in acquisition times of these two sources are discussed in Chapter 5. Afterward, these pairs of matrices were subjected to a data differencing operation to derive the change maps. Sentinel-3 change maps were produced for both the Chl-a and TSM variables. The Sentinel-3 change maps were plotted and displayed using Matplotlib and Basemap Python libraries, much like was done for the HYPSON-1 change maps [68] [69]. Matplotlib was also utilized to generate scales for the Sentinel-3 change maps, representing the magnitude of change in units of Δmgm^{-3} for chlorophyll (Chl-a) or Δgm^{-3} for total suspended matter concentration. These scales provided a visual representation of the extent of change in the respective variables within the Sentinel-3 change maps.

Chapter 4

Results

4.1 Overview

The results section presents the results from the ICA-based change detection techniques applied to the HYPSON-1 hyperspectral time series datasets. The primary objective was to visualize change of detected anomalies over a specific bitemporal time frame using independent components from ICA. The analysis involved five distinct ICA-based change detection techniques: Spatial Stacking, Spectral Stacking, Spectral Differencing, Spectral Difference Stacking, and Component Matching. Additionally, Component Matching used three separate ICA initialization routines to explore different strategies for initialing the un-mixing matrix.

For each technique, change maps were generated to identify areas with changing detected anomalies at the various observation targets. These change maps provide information regarding the locations and distributions of detected changes. The change maps were also accompanied by relative reflectance spectra recovered from the mixing and un-mixing. The spectra allow the independent components and change maps to be identified and characterized by their spectral signatures. For the ICA-based change detection techniques where it is necessary, the relative reflectance spectra have been divided and displayed as before and after spectra. For other techniques, such as Spectral Differencing, a single relative reflectance spectrum exists for each change map.

Due to the volume of results, a subset of two change maps from each technique is presented in this section. This was repeated for each of the four observational target sites (Section 3.3). The selection of change maps was based on their resemblance to algae, sediments, or other potentially relevant detected anomalies in terms of the spatial distribution or spectral signatures. Some of them were selected after manually comparing them using the validation data from the NOAA HAB Forecast and Sentinel-3 OLCI products (Section 3.7). The comparison with validation sources is covered in the discussion chapter (Chapter 5). The approach of including a subset of the change maps ensures that the results are concise and focused on relevant detected anomalies.

4.2 Interpretation of Change Detection Plots

The analysis of the HYPSON-1 hyperspectral data involved the application of five unique ICA-based change detection techniques. Each of these techniques generated sets of independent component spatial maps and sets of relative reflectance spectra. The exact number of sets of independent components and spectra produced by each change detection technique is summarized in Table 5.11 in Chapter 5. The independent components were used to derive and plot change maps, indicating spatial distribution of changing features. The relative reflectance spectra were plotted separately as ICA weights, measured in ambiguous units.

It is important to acknowledge that the change maps and relative reflectance spectra plots are interpreted with slight variations based on the change detection technique used to create them. The following subsections elaborate on how the change maps and spectra produced by each change detection technique are displayed and should be interpreted.

Spatial Stacking

The spatial stacking technique generates two sets of independent component maps, corresponding to the before and after captures. A single set of change maps is derived from these independent component maps through data differencing. As both the before and after captures are processed by the same instance of ICA, their independent components (ICs) share the same relative reflectance spectra. Consequently, the relative reflectance plots for both captures are identical.

In the spatial stacking technique results, the before capture independent component maps, after capture independent component maps, and change maps are plotted. Additionally, the relative reflectance spectra are plotted alongside the before and after independent component maps.

Spectral Stacking

The spectral stacking technique generates a single set of independent component maps from the stacked composite datacube. The independent component maps are directly interpreted as change maps. As the stacked composite datacube has double the number of spectral bands, relative reflectance spectra are split in half. This produces two different sets of relative reflectance spectrum plots, one for the independent components from the before capture and the other for the independent components from the after capture.

In the spectral stacking technique results, only the change maps are plotted. Each change map is accompanied by an independent component relative reflectance spectrum plot from the before capture and another independent component relative reflectance spectrum plot from the after capture.

Spectral Differencing

The spectral differencing technique generates a single set of independent component maps from the differential datacube. The independent component maps are directly interpreted as change maps. As the differential datacube has the standard the number of spectral bands, a single set of relative reflectance spectra is produced. The relative reflectance spectra are interpreted as differential relative reflectance spectra, describing the change between the before and after captures.

In the spectral differencing technique results, only the change maps are plotted. Each change map is accompanied by a differential independent component relative reflectance spectrum plot.

Spectral Differencing and Stacking Hybrid

The spectral differencing and stacking hybrid technique generates a single set of independent component maps from the composite datacube, formed from stacking the before capture datacube and a differential datacube. The independent component maps are directly interpreted as change maps. As the composite datacube has double the number of spectral bands, relative reflectance spectra are split in half. This produces two different sets of relative reflectance plots, one for the before capture and the other for the differential datacube. The spectra associated with the differential datacube are interpreted as differential relative reflectance spectra, describing the change between the before and after captures.

In the spectral differencing and stacking hybrid technique results, only the change maps are plotted. Each change map is accompanied by an independent component relative reflectance spectrum plot from the before capture and a differential independent component relative reflectance spectrum plot.

Component Matching

The component matching technique processes each of the before and after captures independently by ICA. It generates two sets of independent component maps, each corresponding to the before and after captures. Additionally, two different sets of relative reflectance spectra are also produced, again corresponding to the before and after captures. A single set of change maps is derived from the independent component maps through data differencing.

In the component matching technique results, the before independent component maps, after independent component maps, and change maps are plotted. Additionally, the before and after independent component relative reflectance spectra are plotted alongside the before and after IC maps. The Pearson correlation coefficient score used to match the independent components to create the change map is included in the title of each component matching figure. Red vertical lines are also added to the relative reflectance spectrum plots to indicate the range of spectral bands used in the Pearson correlation coefficient comparison and matching procedure.

4.3 Lake Erie Change Detection Results

Description

The Lake Erie change detection results are generated from two HYPSON-1 hyperspectral captures acquired on 19 July 2022 15:50 UTC and 27 August 2022 16:05 UTC. The target and HYPSON-1 data are discussed in Section 3.3.

Spatial Stacking Results

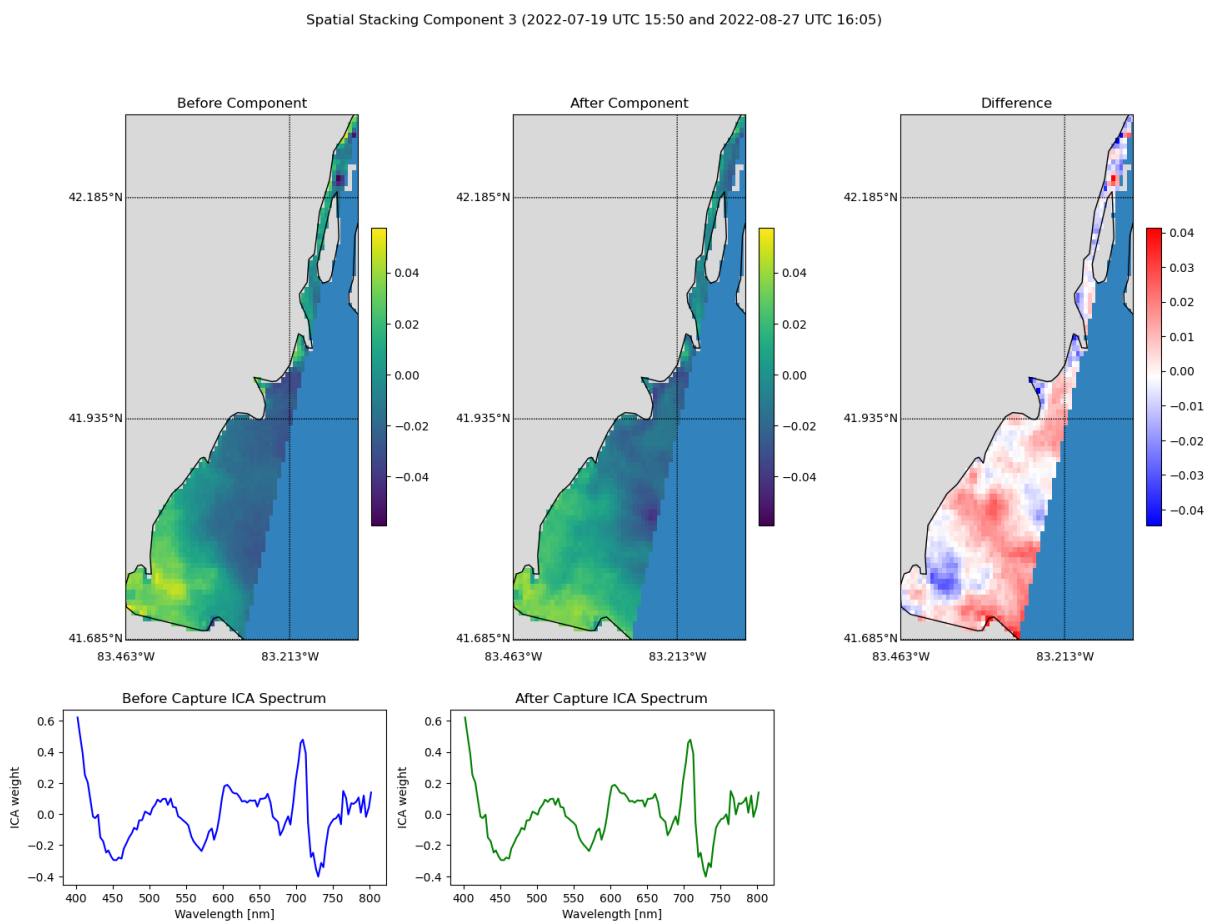


Figure 4.1: Spatial stacking component 3.

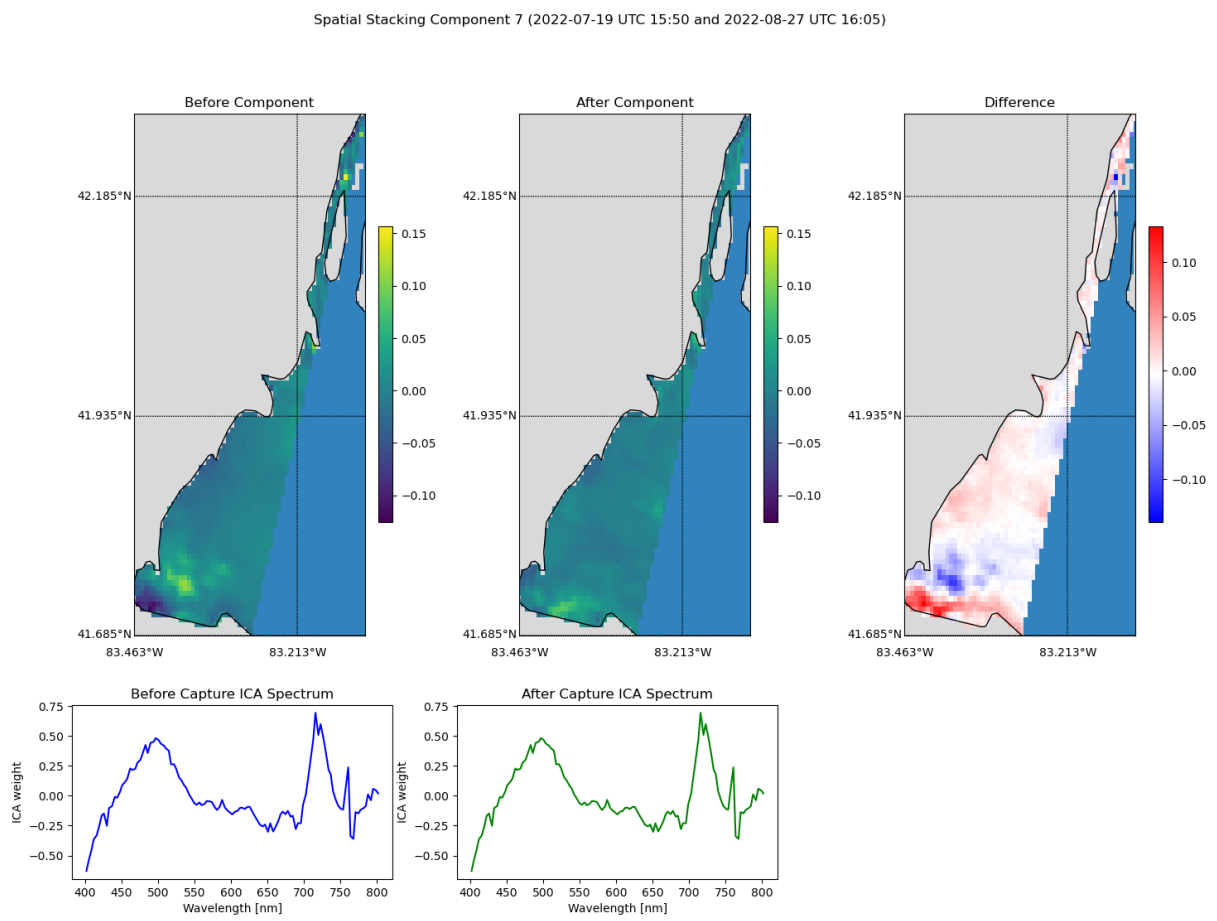


Figure 4.2: Spatial stacking component 7.

Spectral Stacking Results

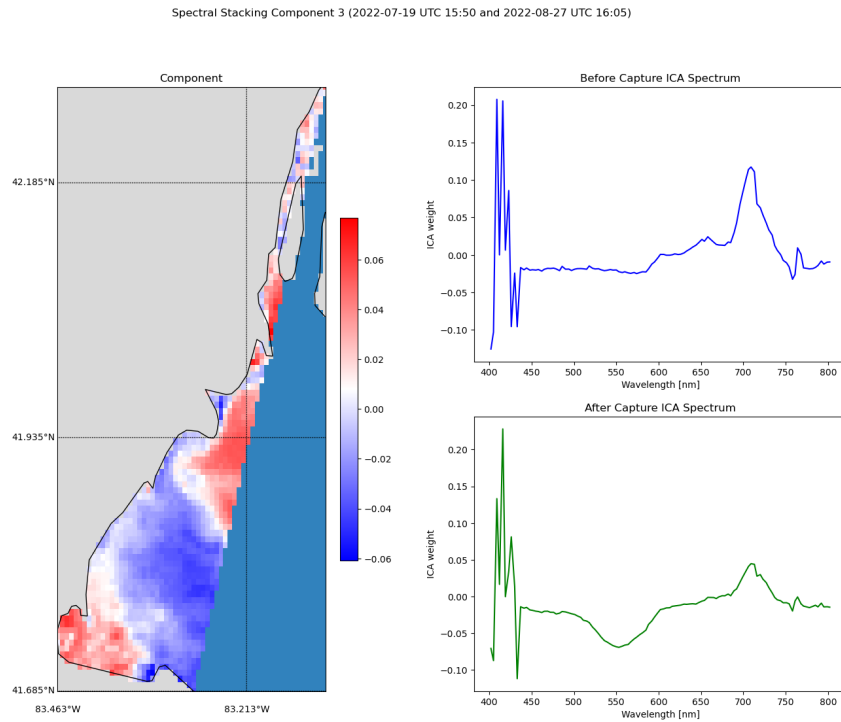


Figure 4.3: Spectral stacking component 3.

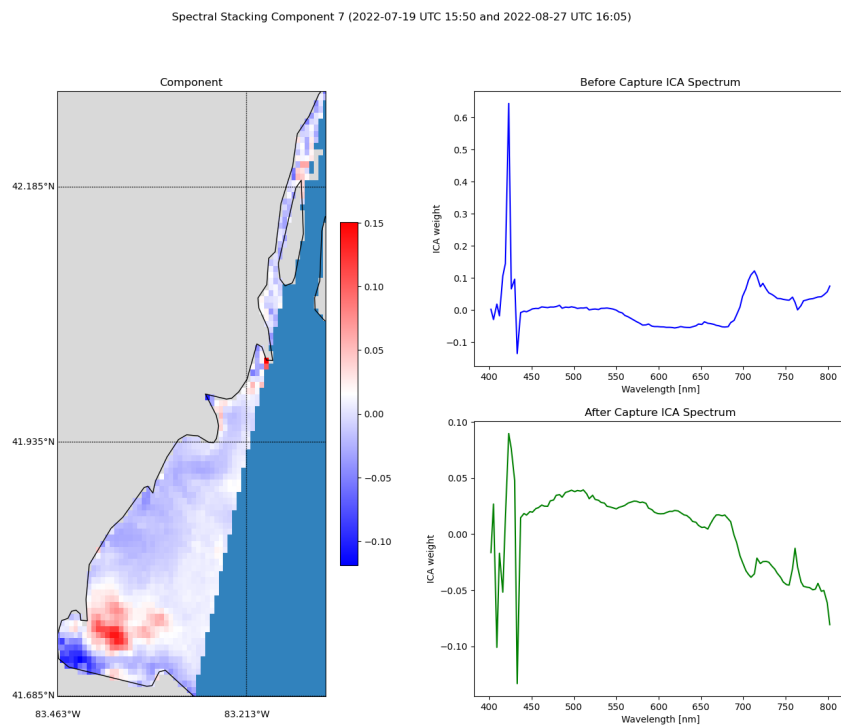


Figure 4.4: Spectral stacking component 7.

Spectral Differencing Results

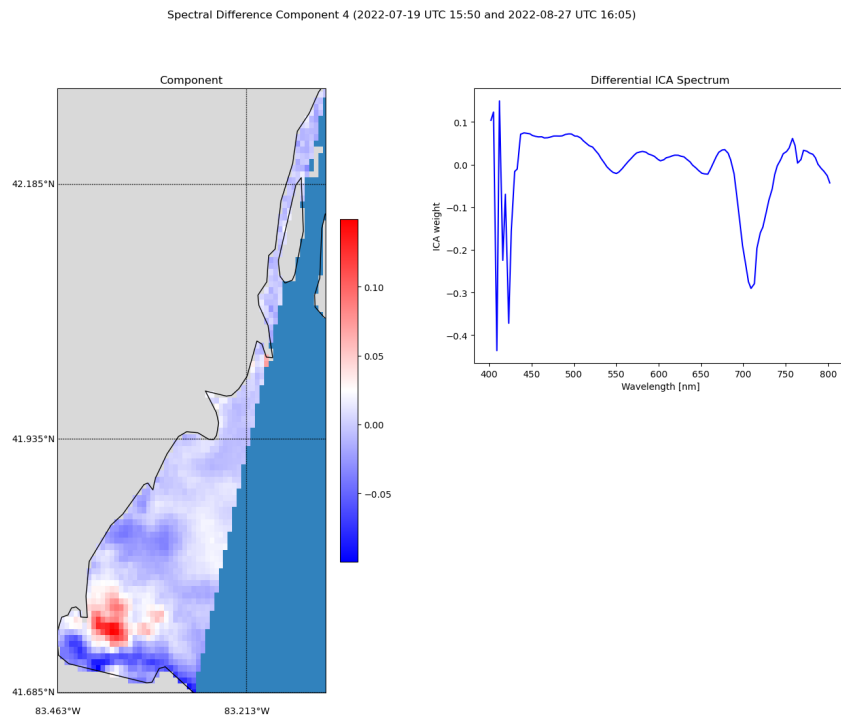


Figure 4.5: Spectral differencing component 4.

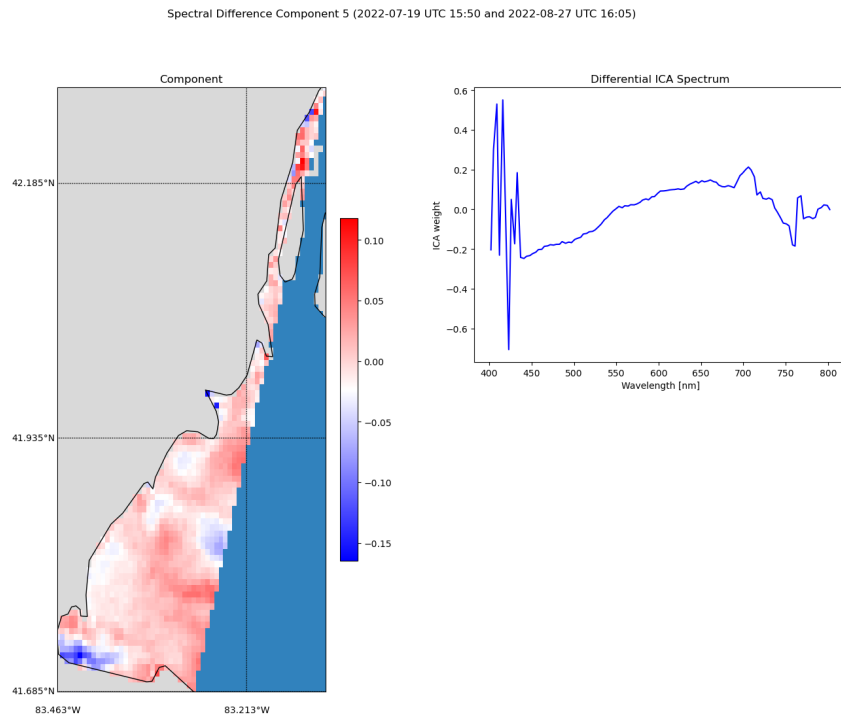


Figure 4.6: Spectral differencing component 5.

Spectral Differencing and Stacking Hybrid Results

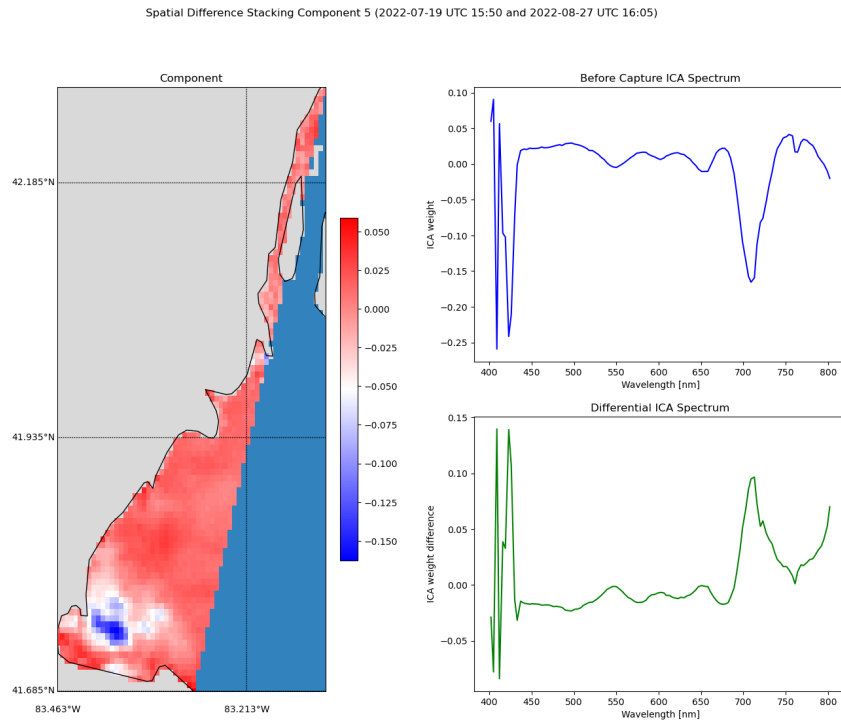


Figure 4.7: Spectral differencing and stacking component 5.

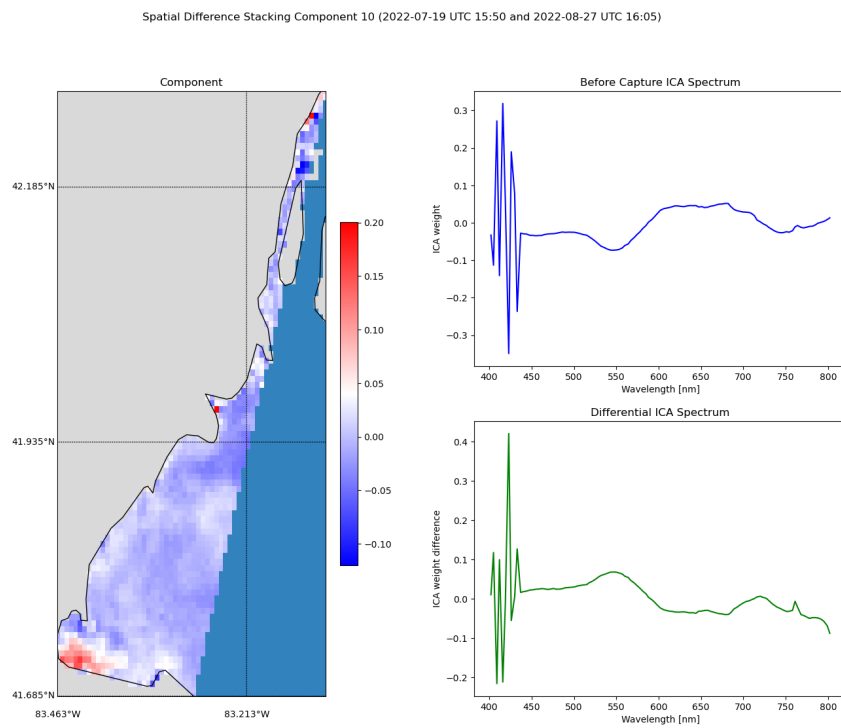


Figure 4.8: Spectral differencing and stacking component 10.

Component Matching Results

Random Un-mixing Matrix

Match: c10 and c7, Score: -0.7790148543812495 (2022-07-19 UTC 15:50 and 2022-08-27 UTC 16:05)

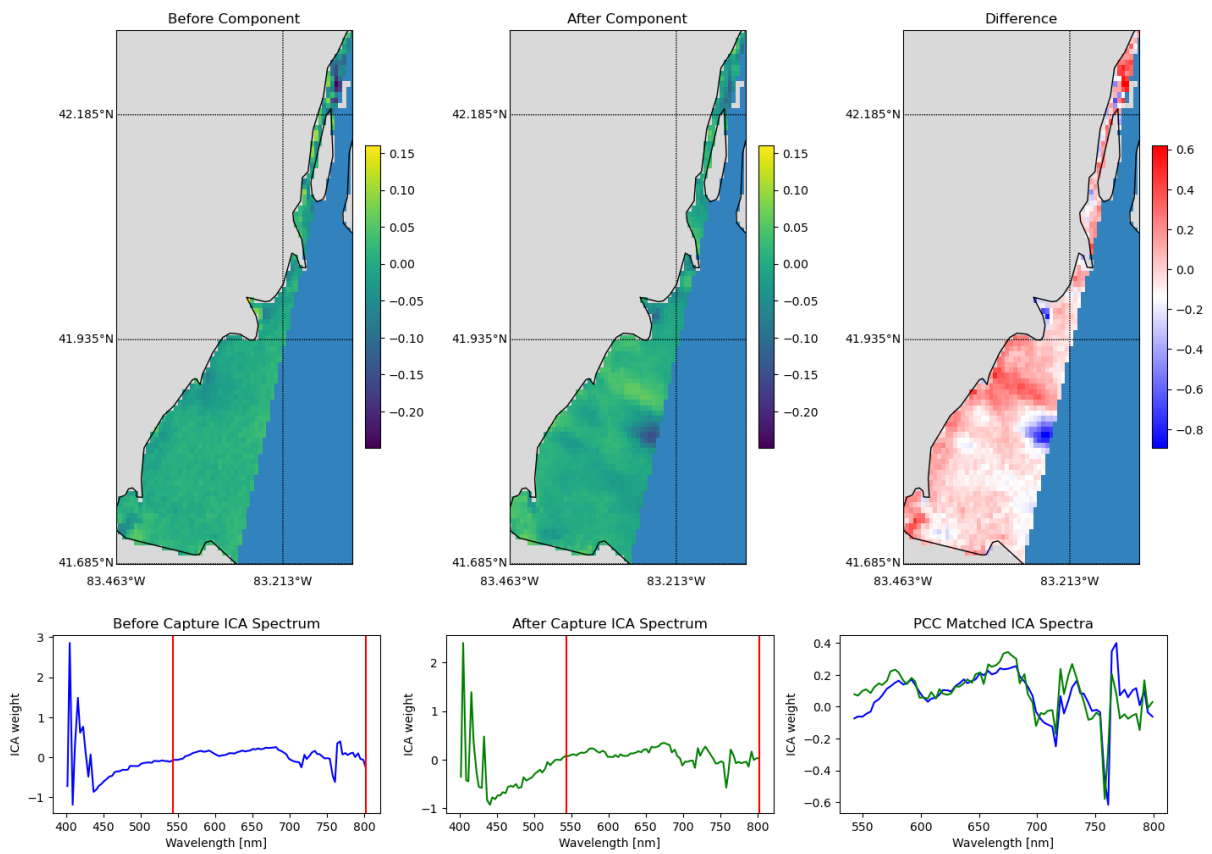


Figure 4.9: Component match 3.

Match: c3 and c2, Score: -0.6364514177032892 (2022-07-19 UTC 15:50 and 2022-08-27 UTC 16:05)

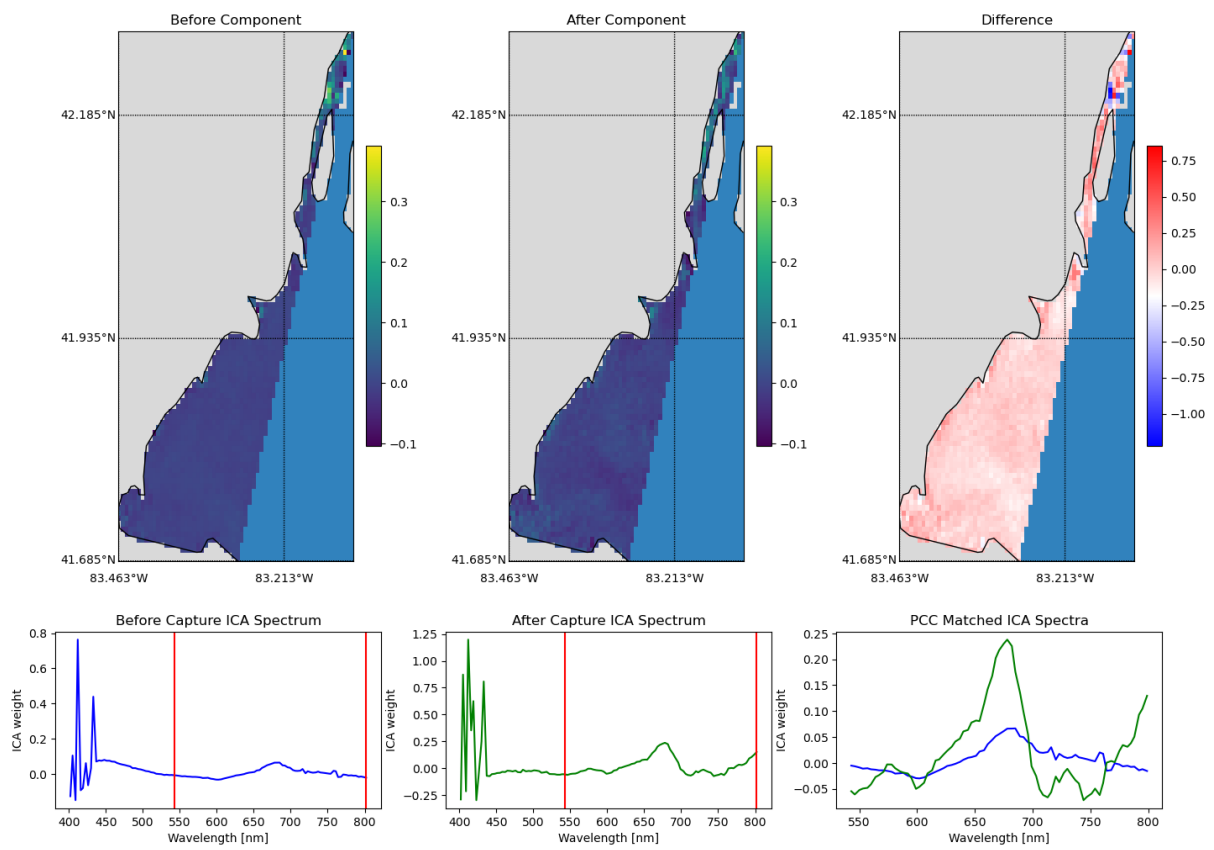


Figure 4.10: Component match 11.

Previous Un-mixing Matrix

Match: c8 and c4, Score: -0.7787641007981875 (2022-07-19 UTC 15:50 and 2022-08-27 UTC 16:05)

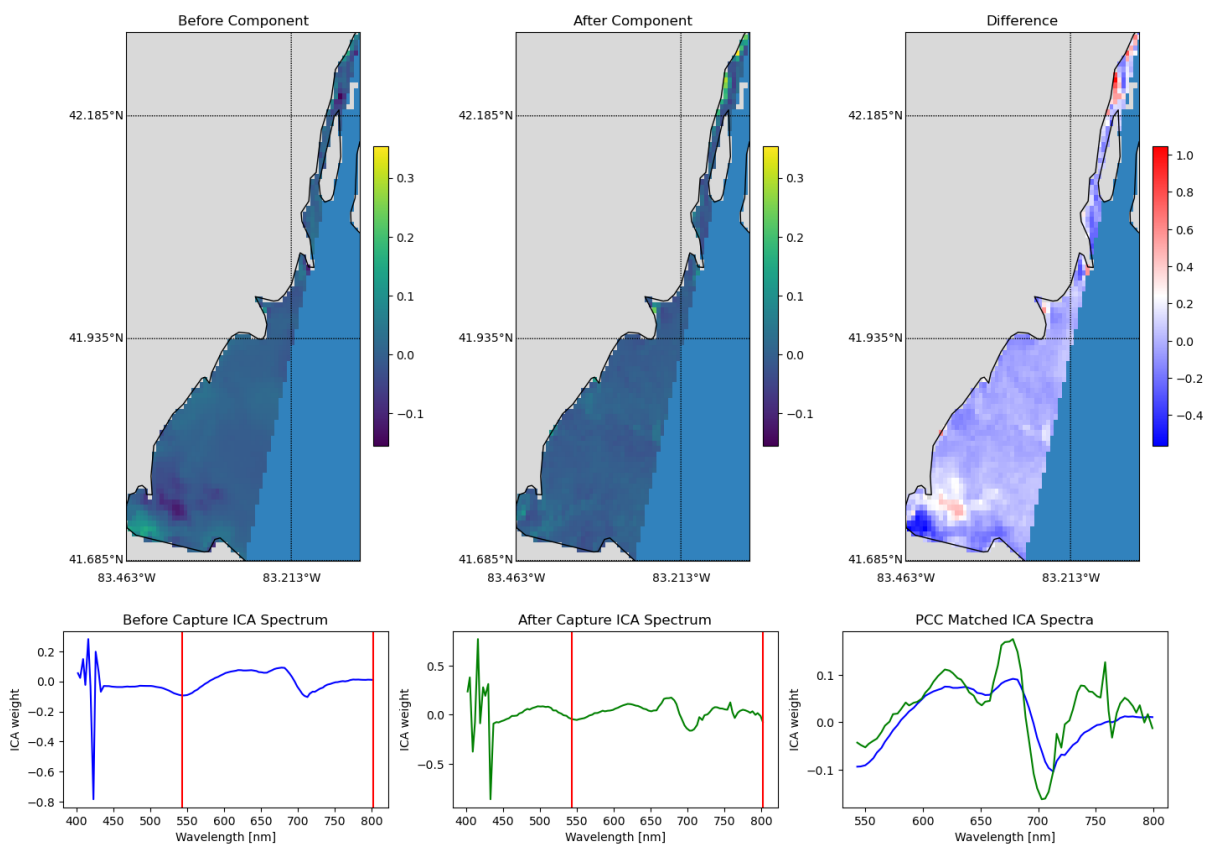


Figure 4.11: Component match 3.

Match: c8 and c9, Score: -0.6424278443381239 (2022-07-19 UTC 15:50 and 2022-08-27 UTC 16:05)

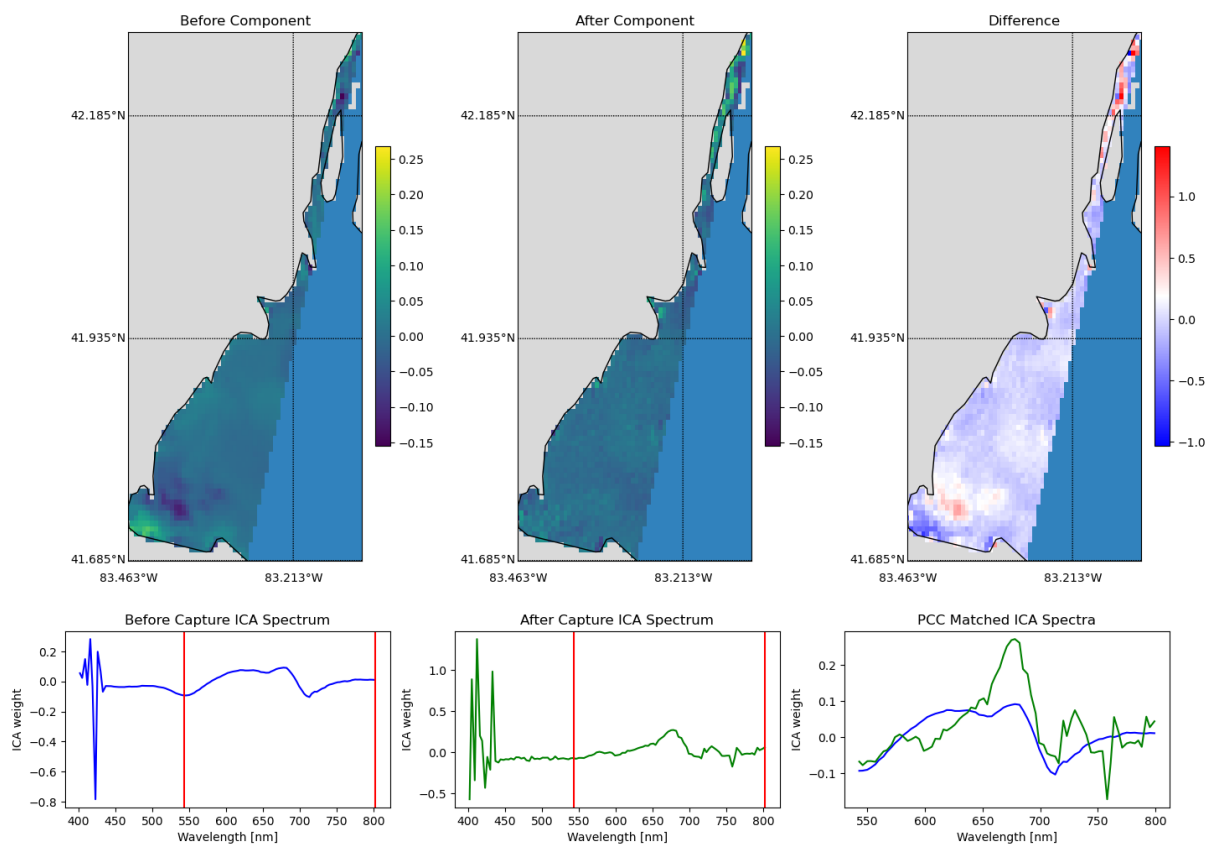


Figure 4.12: Component match 12.

ATGP-generated Un-mixing Matrix

Match: c9 and c2, Score: -0.944833689384501 (2022-07-19 UTC 15:50 and 2022-08-27 UTC 16:05)

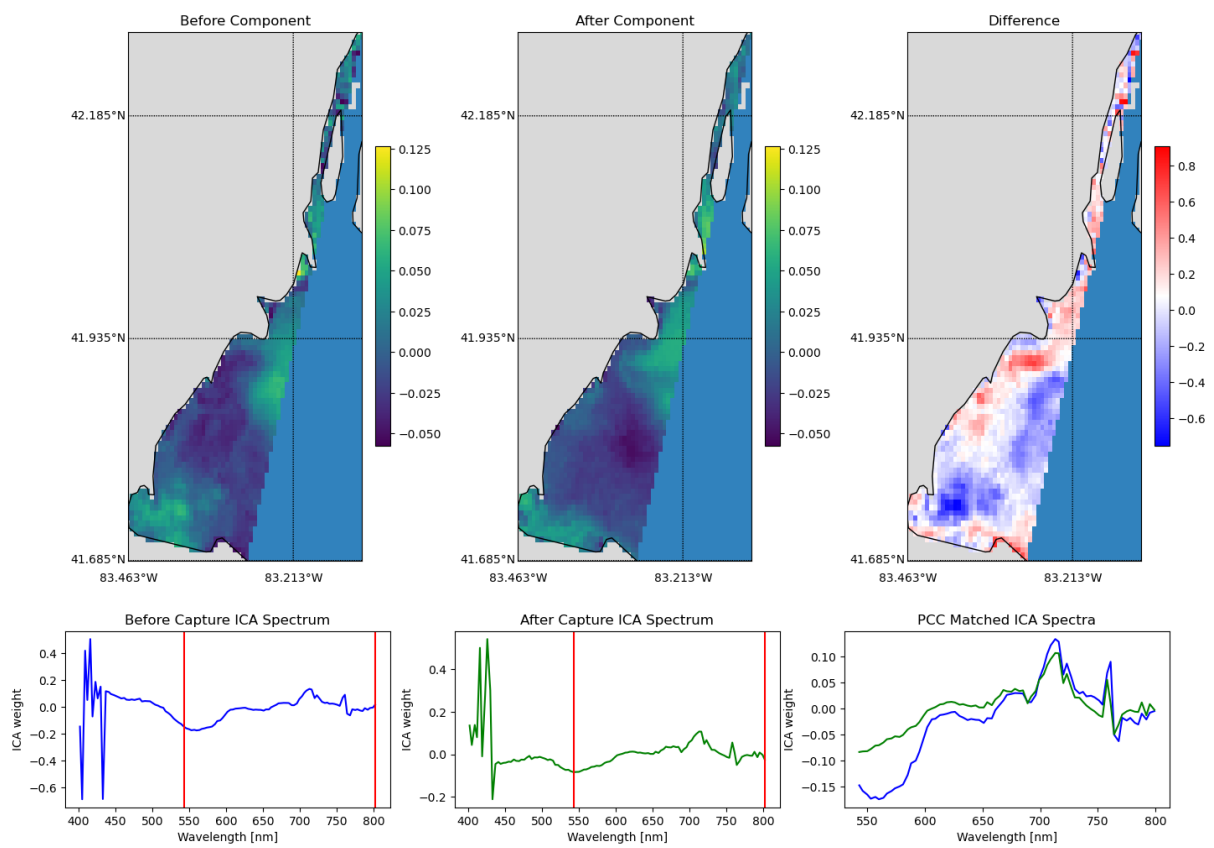


Figure 4.13: Component match 1.

Match: c8 and c8, Score: -0.7769353877463425 (2022-07-19 UTC 15:50 and 2022-08-27 UTC 16:05)

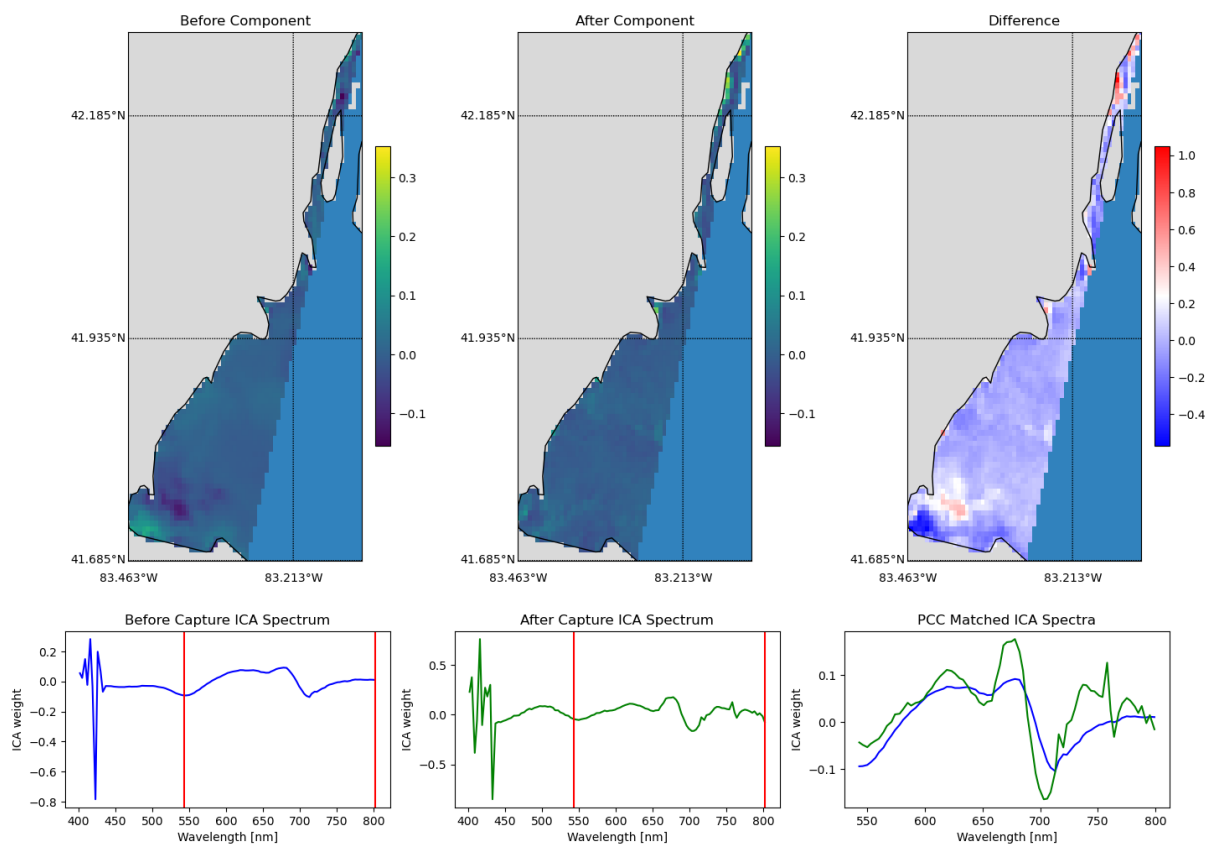


Figure 4.14: Component match 4.

4.4 Salish Sea Change Detection Results

Description

The Salish Sea change detection results are generated from two HYPSON-1 hyperspectral captures acquired on 12 July 2022 18:46 UTC and 13 July 2022 18:34 UTC. The target and HYPSON-1 data are discussed in Section 3.3.

Spatial Stacking Results

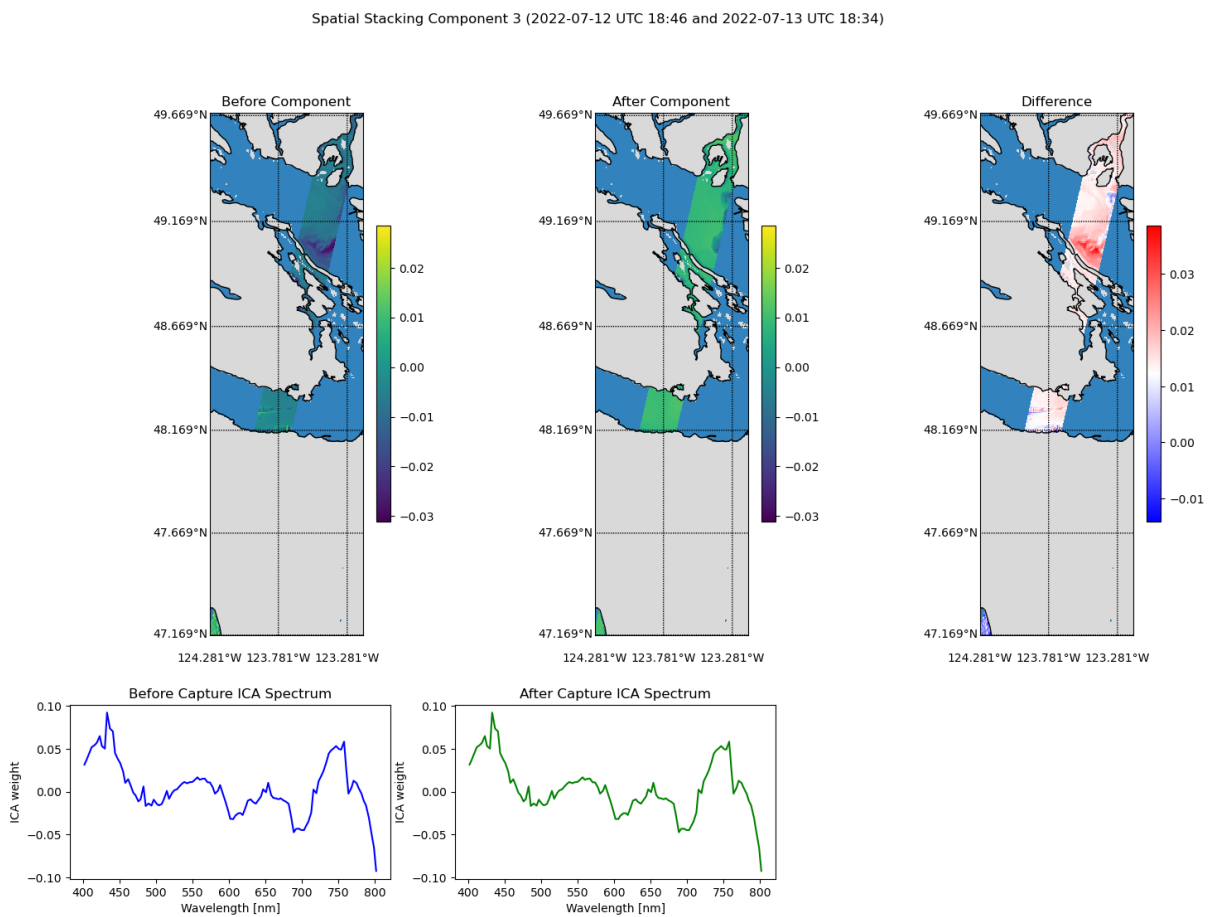


Figure 4.15: Spatial stacking component 3.

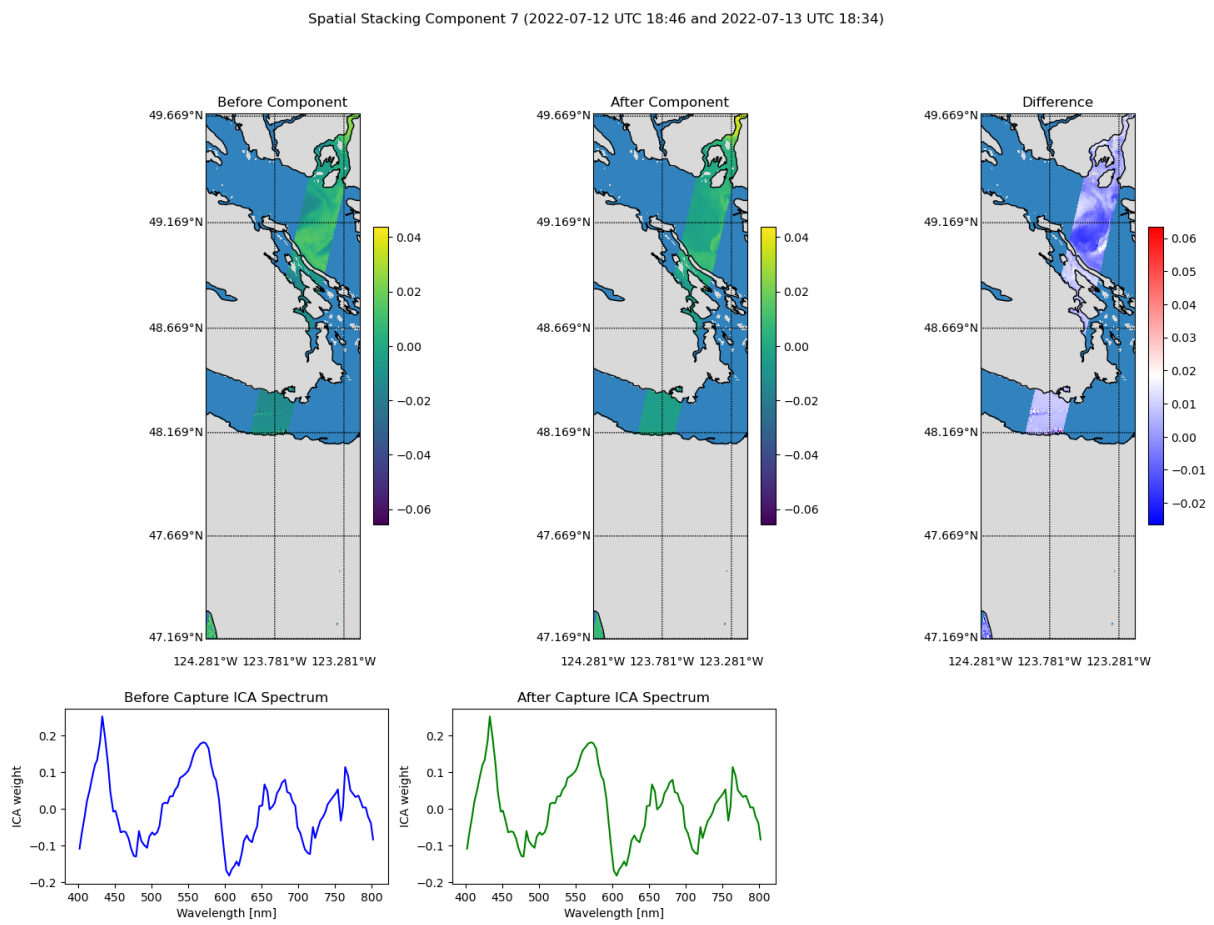


Figure 4.16: Spatial stacking component 7.

Spectral Stacking Results

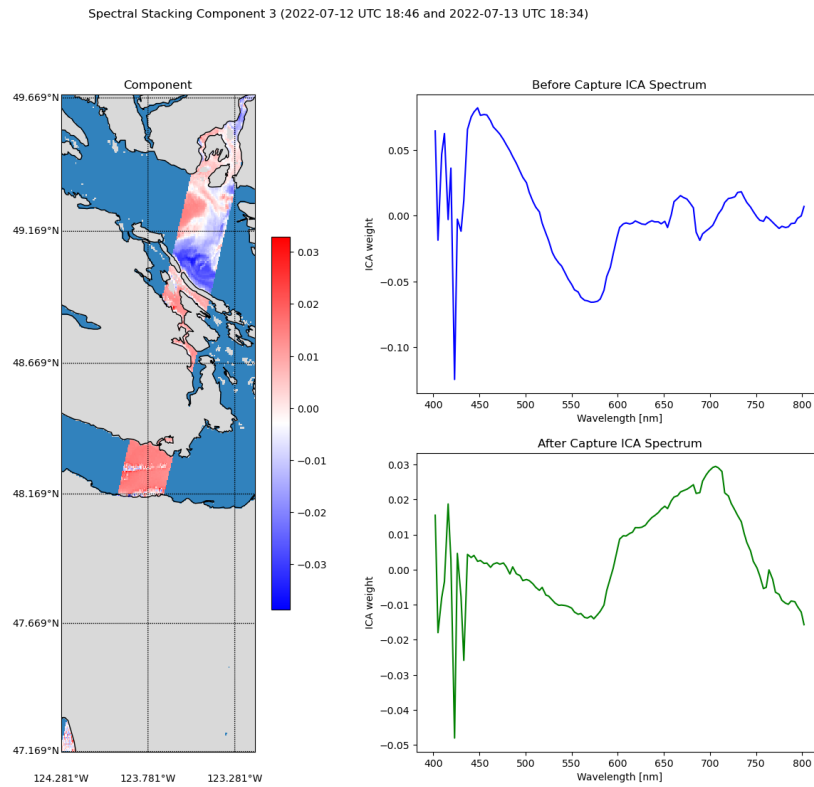


Figure 4.17: Spectral stacking component 3.

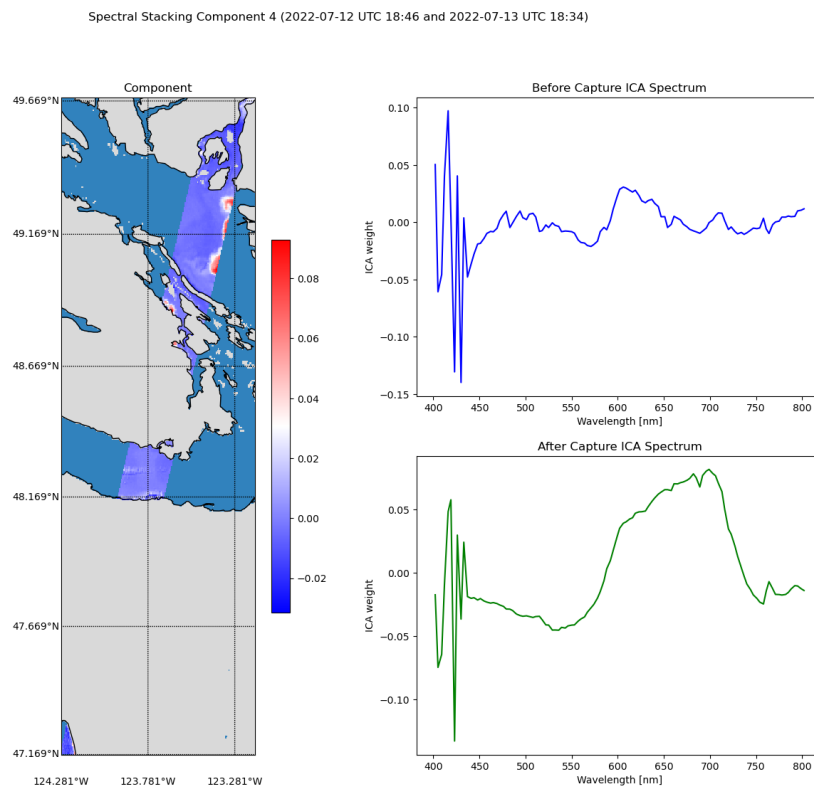


Figure 4.18: Spectral stacking component 4.

Spectral Differencing Results

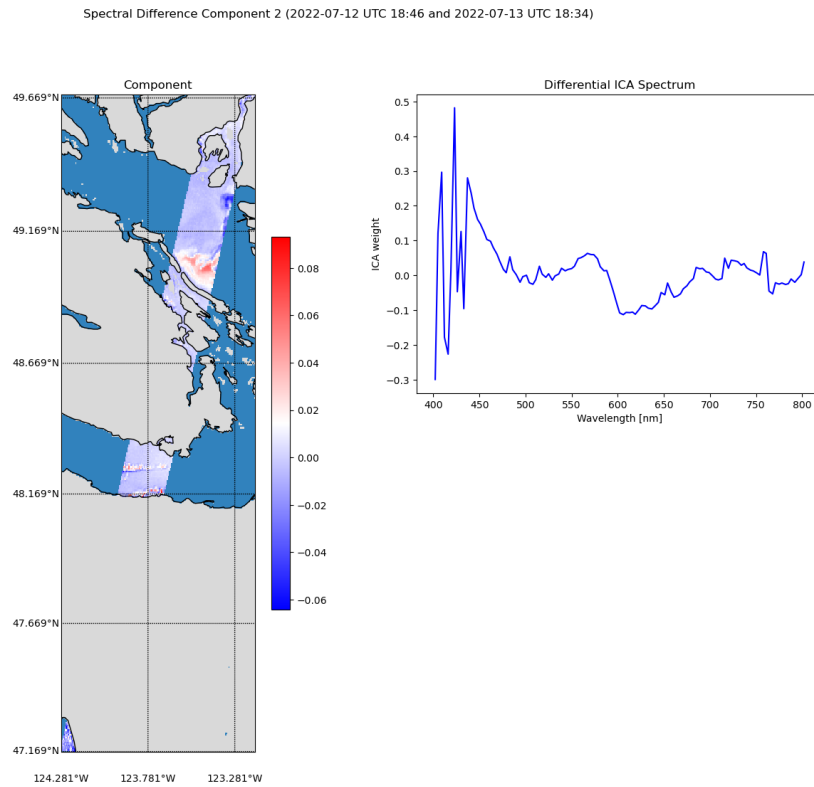


Figure 4.19: Spectral differencing component 2.

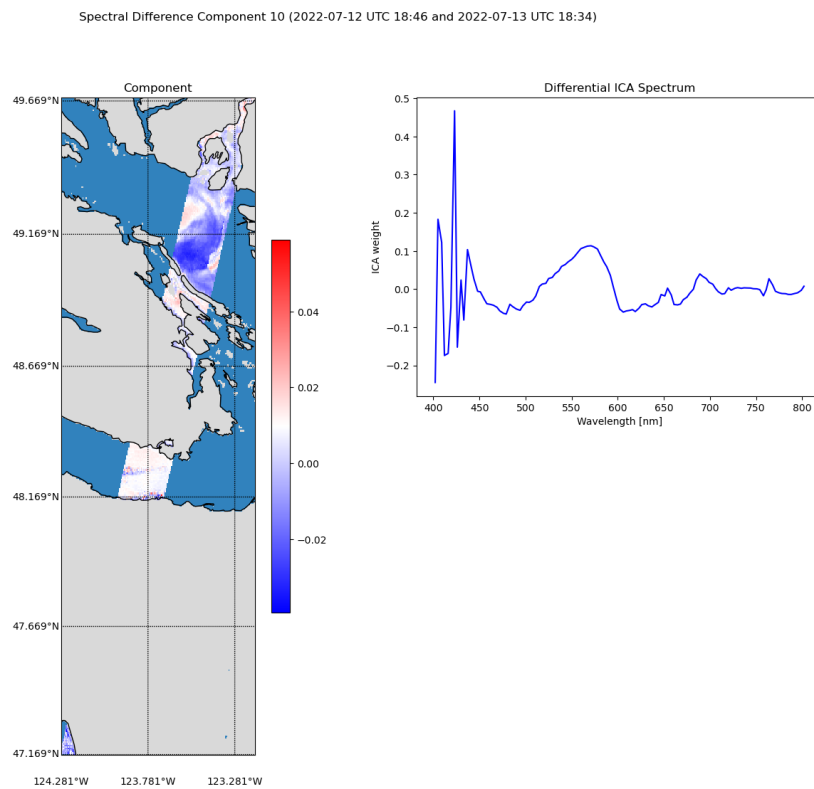


Figure 4.20: Spectral differencing component 10.

Spectral Differencing and Stacking Hybrid Results

Spatial Difference Stacking Component 5 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

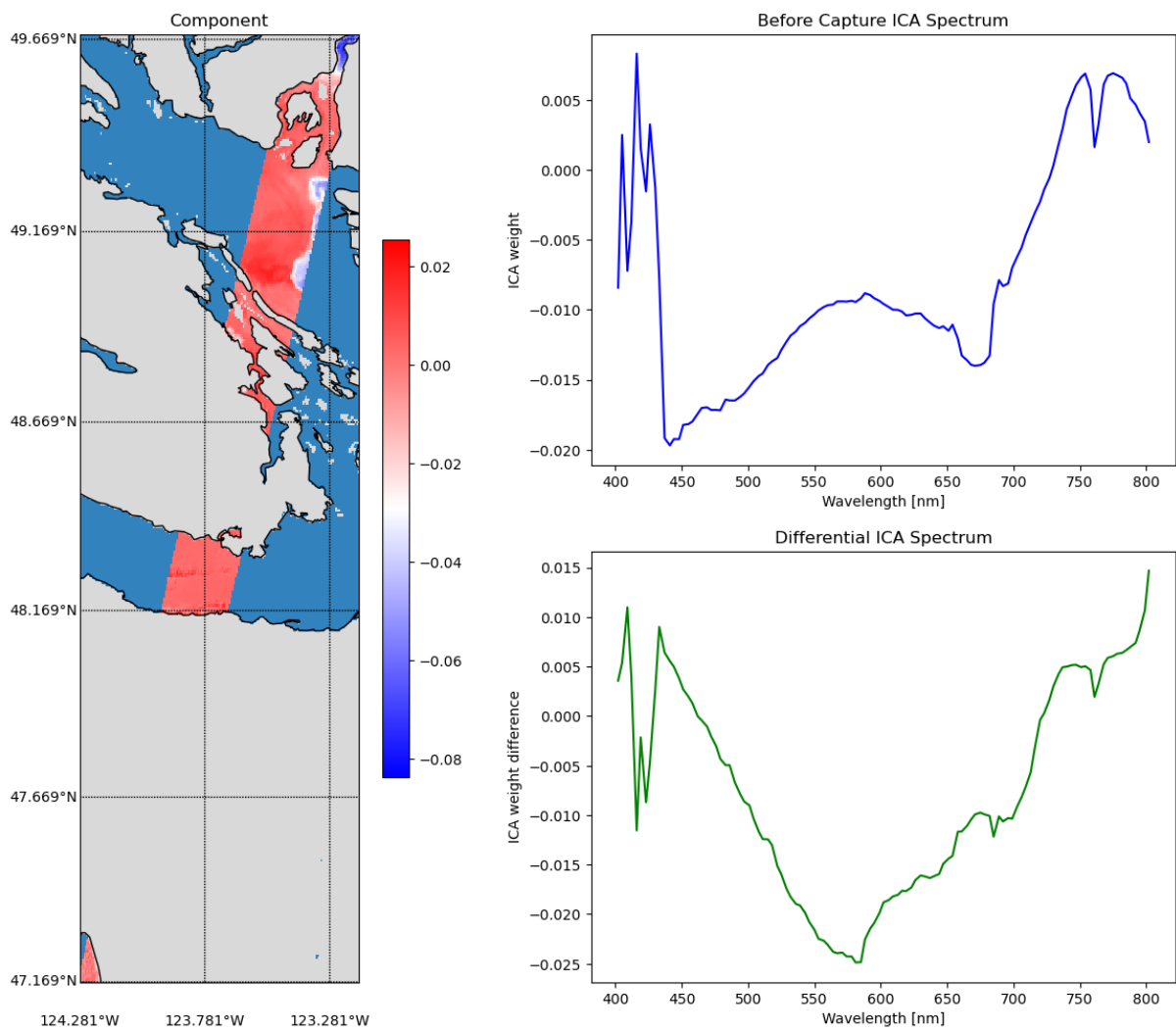


Figure 4.21: Spectral differencing and stacking component 5.

Spatial Difference Stacking Component 10 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

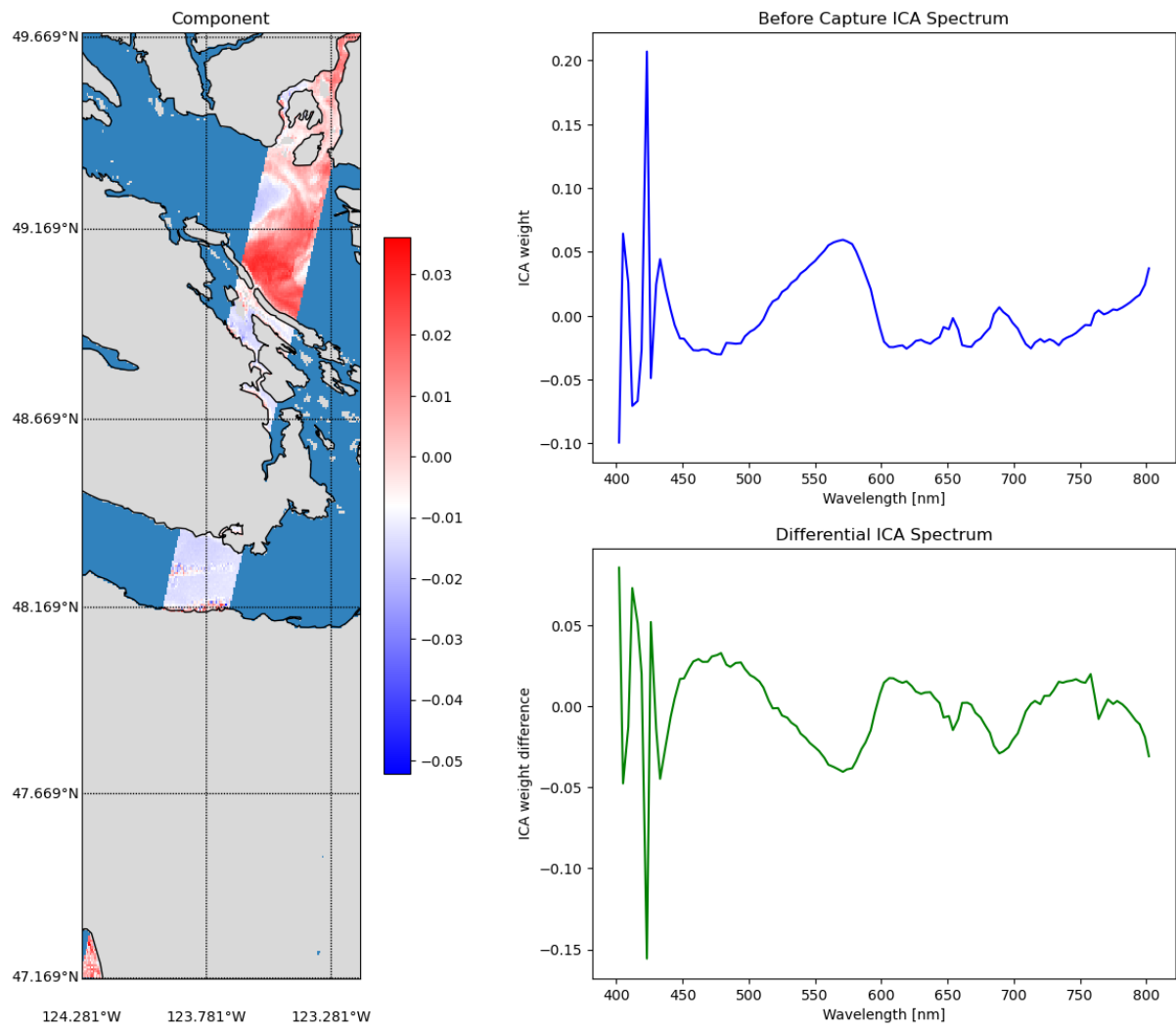


Figure 4.22: Spectral differencing and stacking component 10.

Component Matching Results

Random Un-mixing Matrix

Match: c8 and c5, Score: -0.728307683408084 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

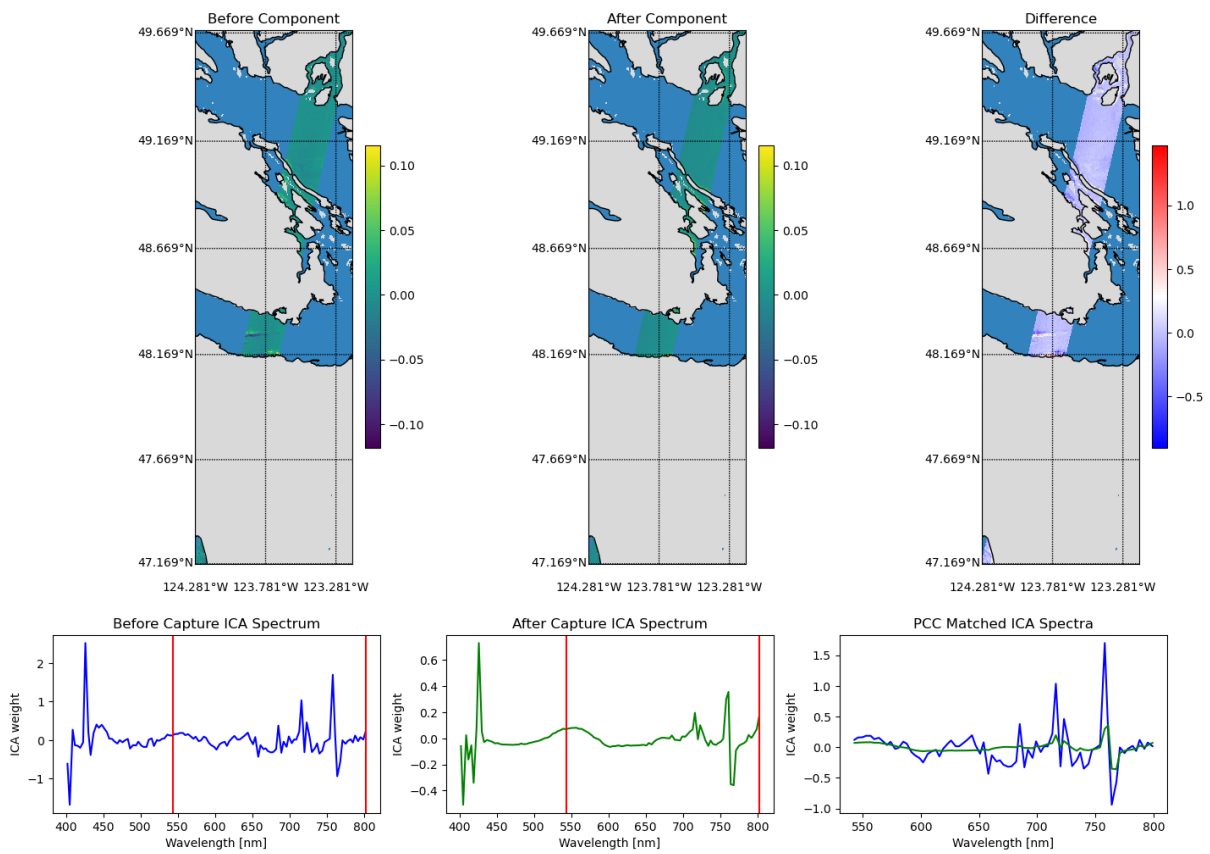


Figure 4.23: Component match 5.

Match: c7 and c5, Score: -0.6592116872647561 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

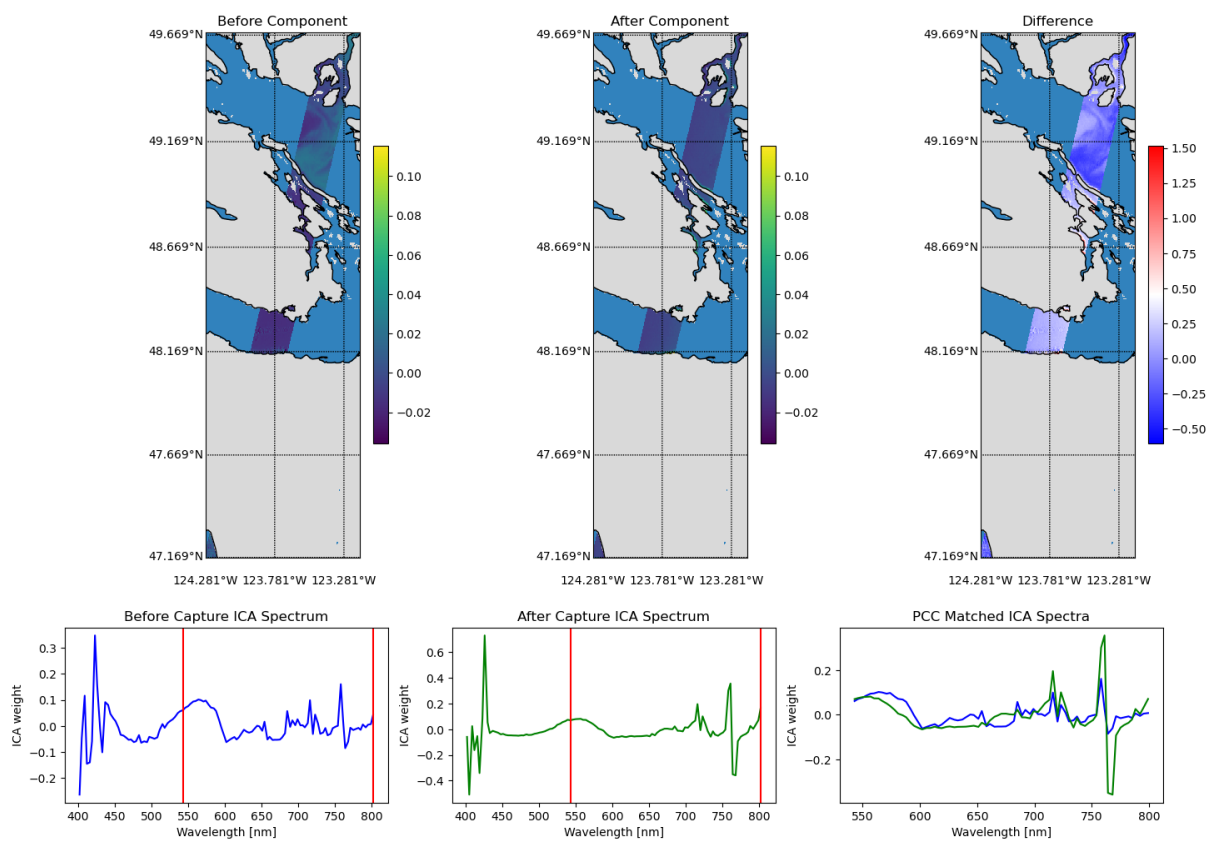


Figure 4.24: Component match 8.

Previous Un-mixing Matrix

Match: c10 and c6, Score: 0.7183042894642064 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

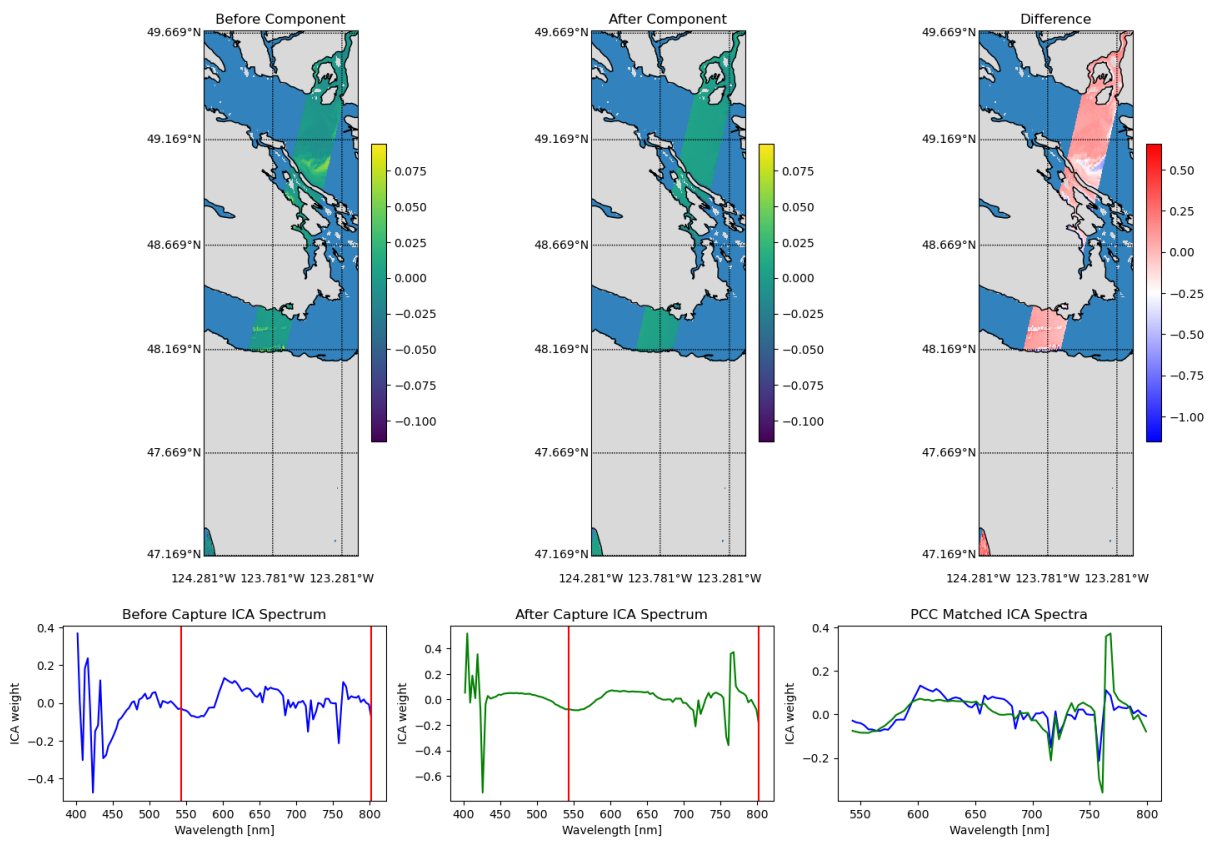


Figure 4.25: Component match 5.

Match: c10 and c7, Score: -0.61603742724667 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

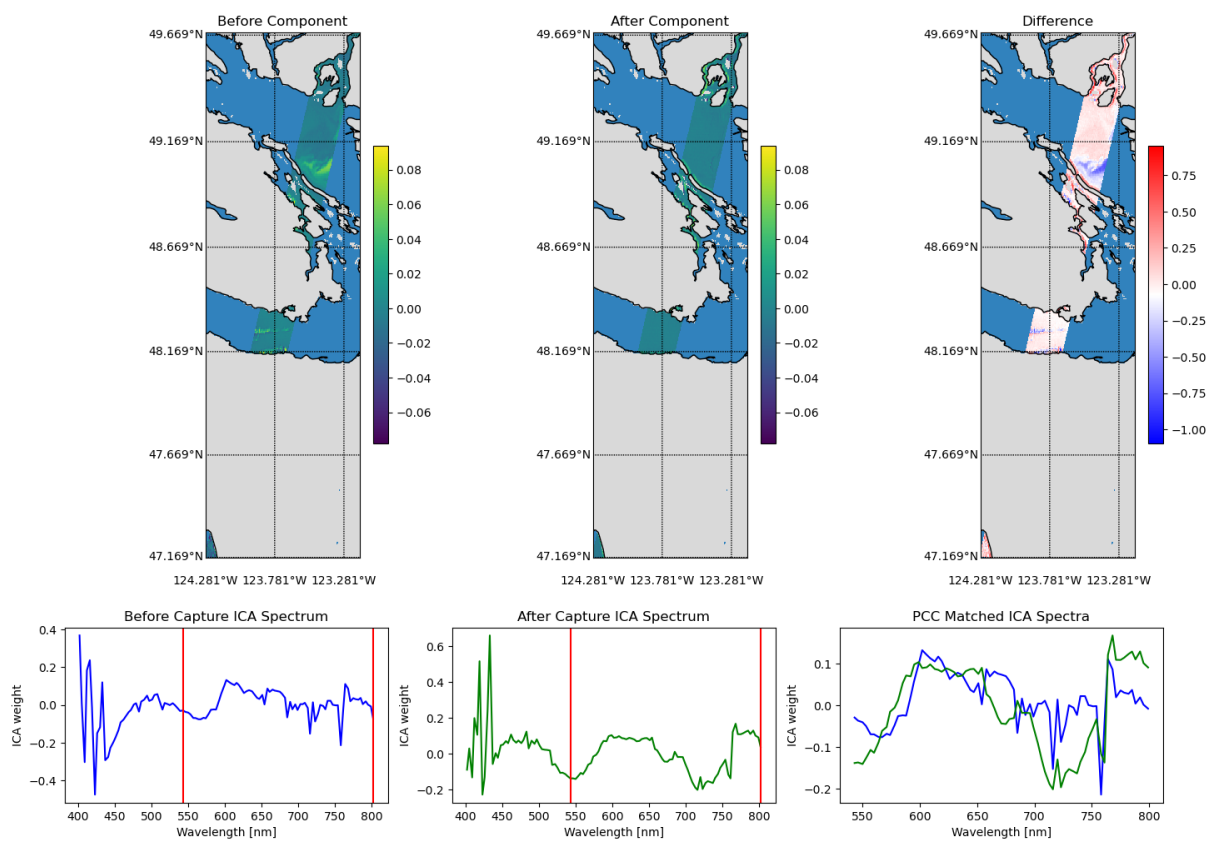


Figure 4.26: Component match 10.

ATGP-generated Un-mixing Matrix

Match: c3 and c10, Score: 0.8063156219238337 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

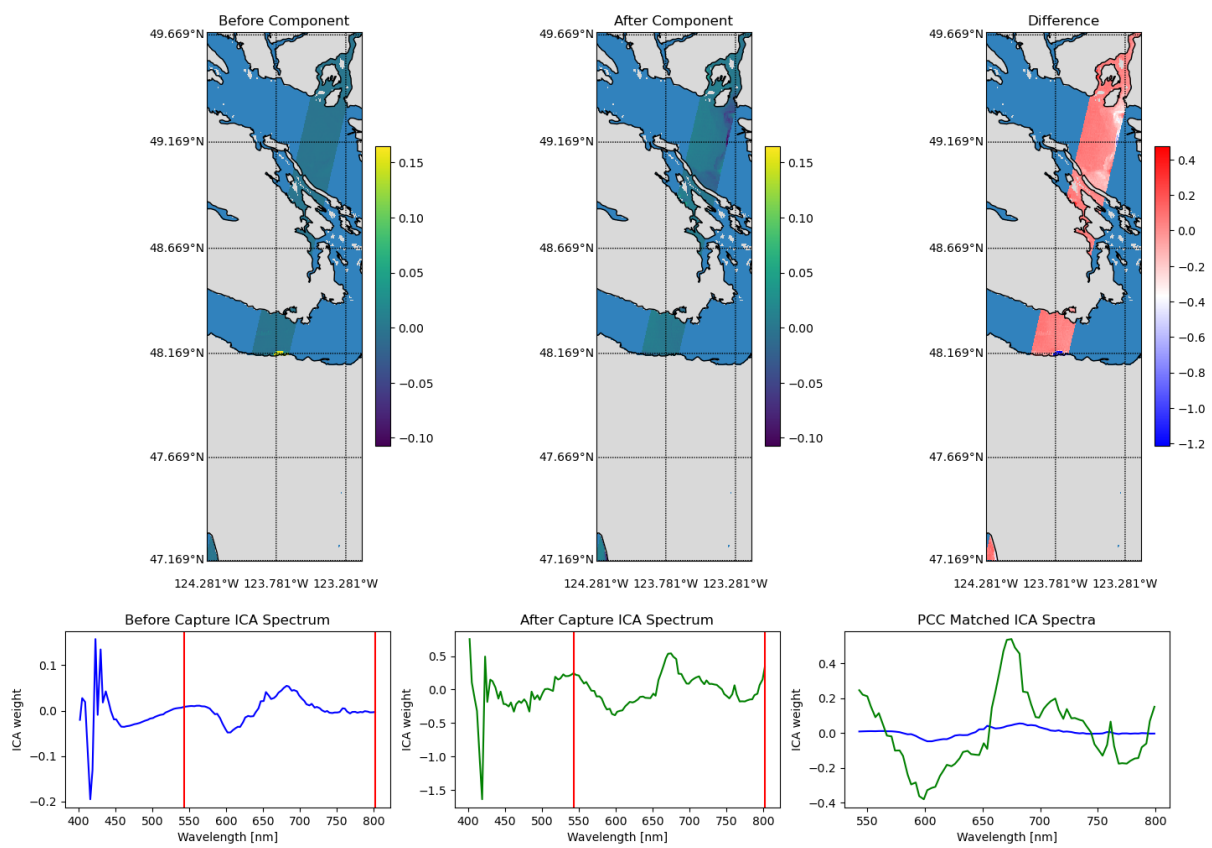


Figure 4.27: Component match 2.

Match: c10 and c5, Score: -0.7246834834908235 (2022-07-12 UTC 18:46 and 2022-07-13 UTC 18:34)

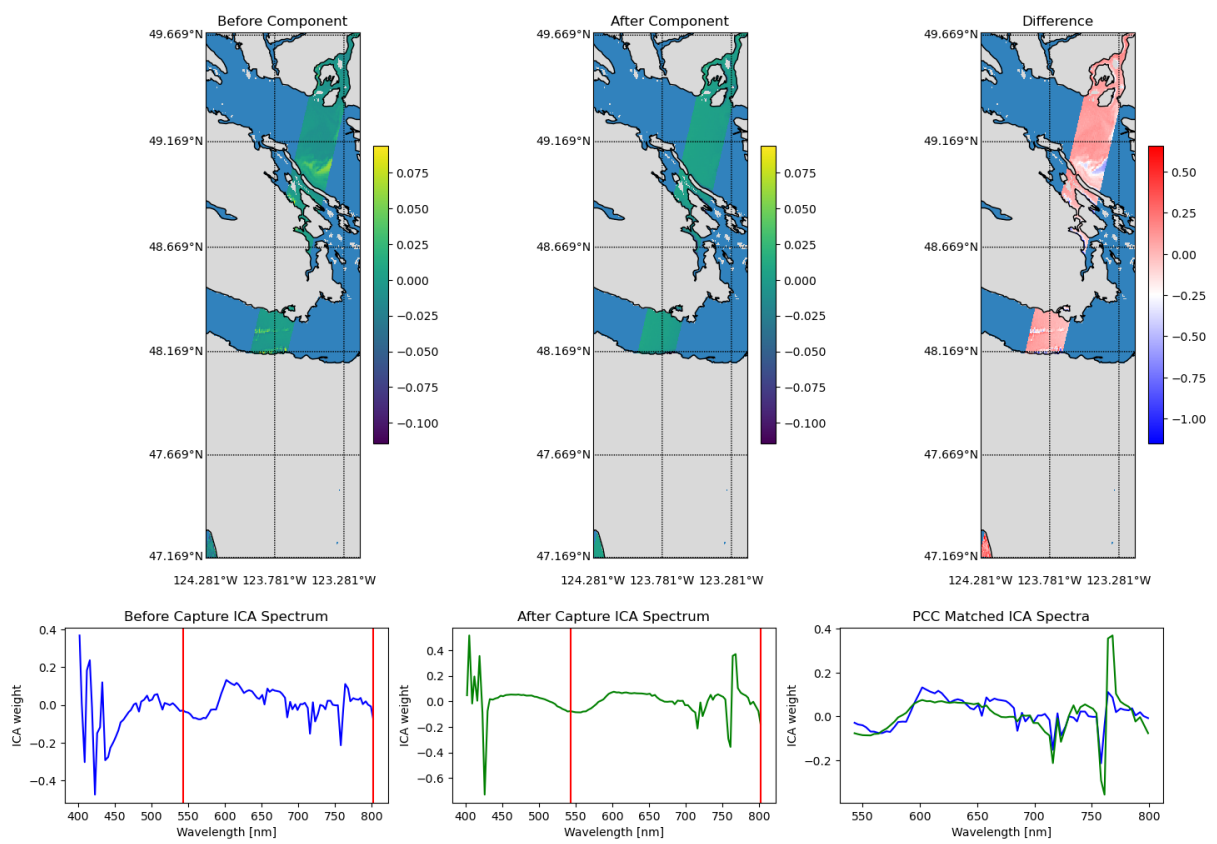


Figure 4.28: Component match 4.

4.5 Danube River Delta Change Detection Results

Description

The Danube River Delta change detection results are generated from two HYPSON-1 hyperspectral captures acquired on 05 March 2023 08:41 UTC and 08 April 2023 08:22 UTC. The target and HYPSON-1 data are discussed in Section 3.3.

Spatial Stacking Results

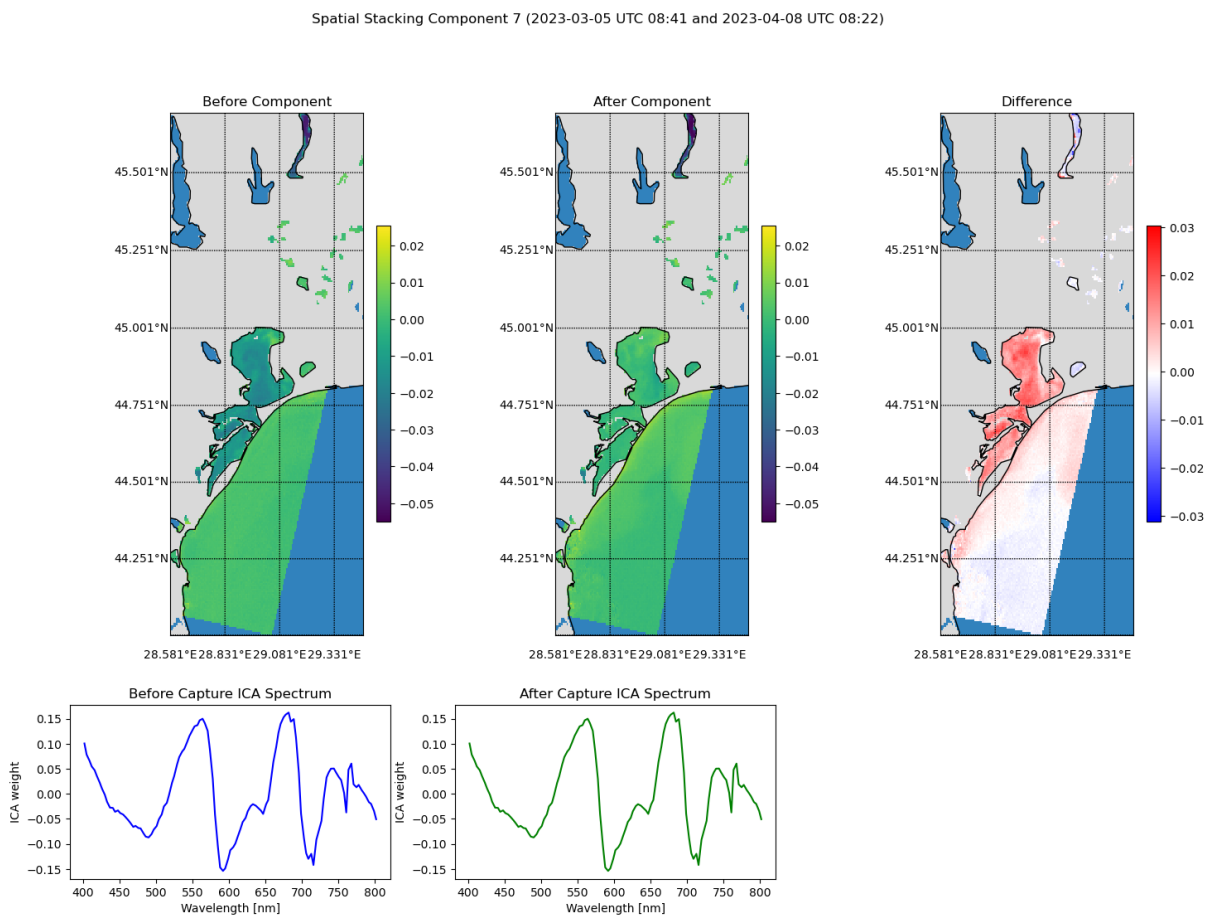


Figure 4.29: Spatial stacking component 7.

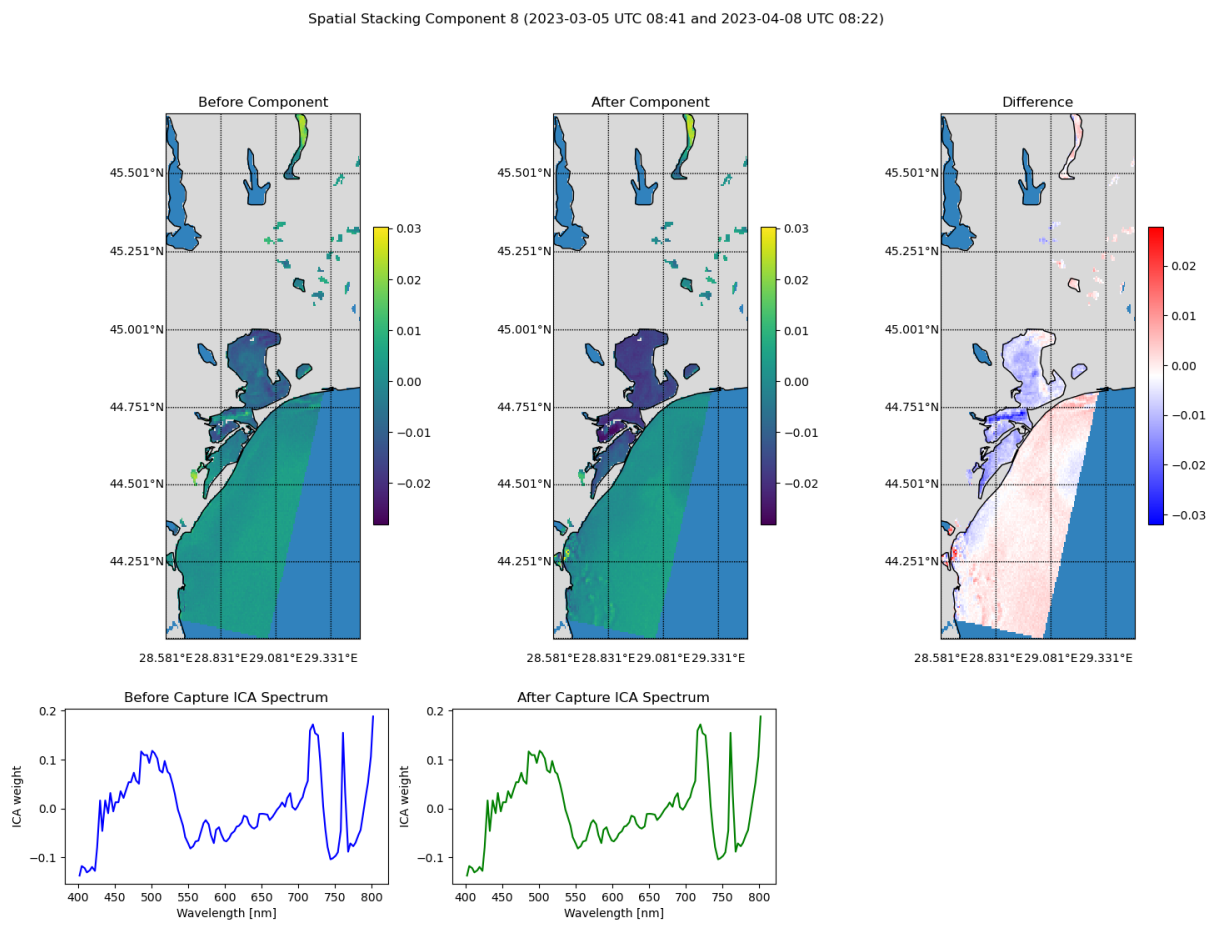


Figure 4.30: Spatial stacking component 8.

Spectral Stacking Results

Spectral Stacking Component 6 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

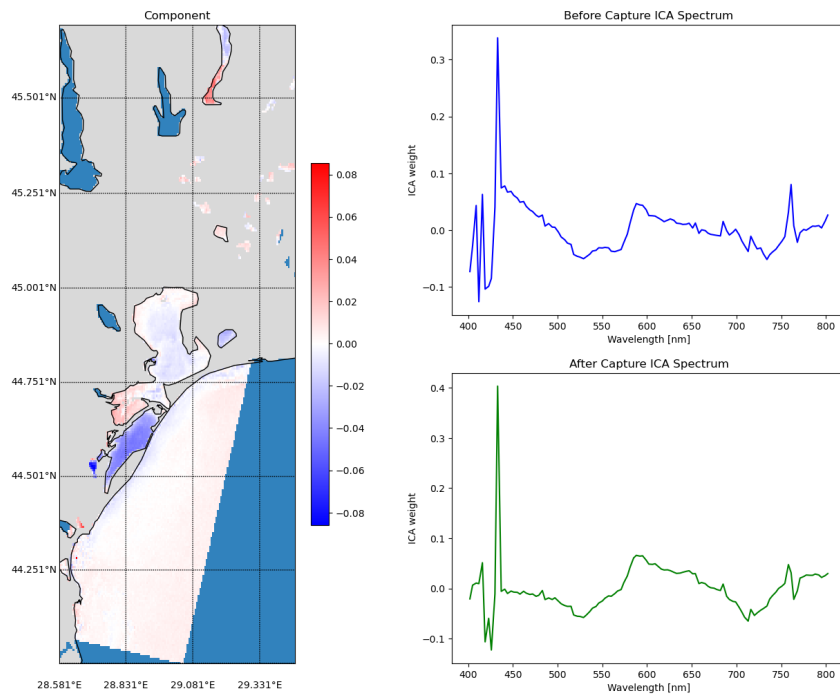


Figure 4.31: Spectral stacking component 6.

Spectral Stacking Component 9 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

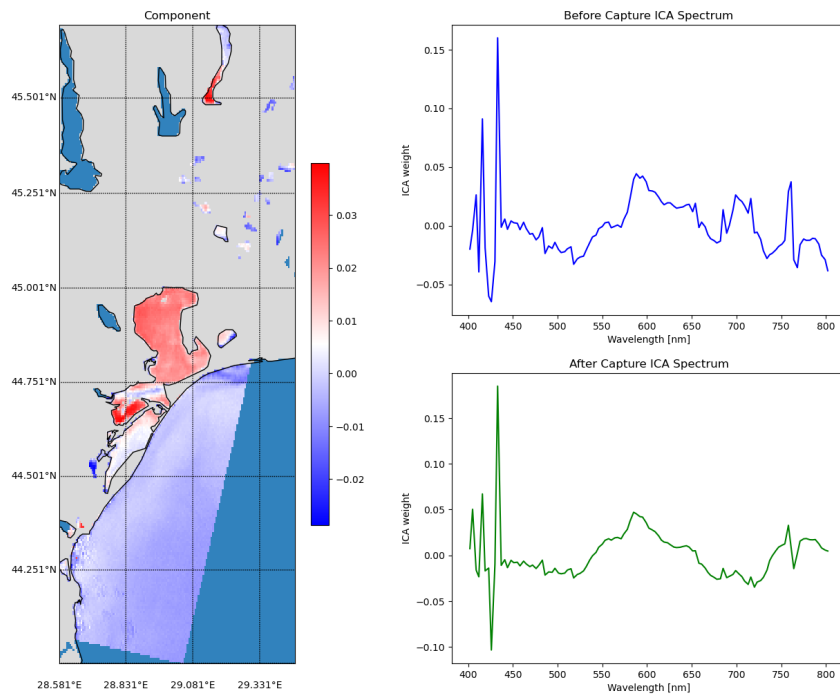


Figure 4.32: Spectral stacking component 9.

Spectral Differencing Results

Spectral Difference Component 3 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

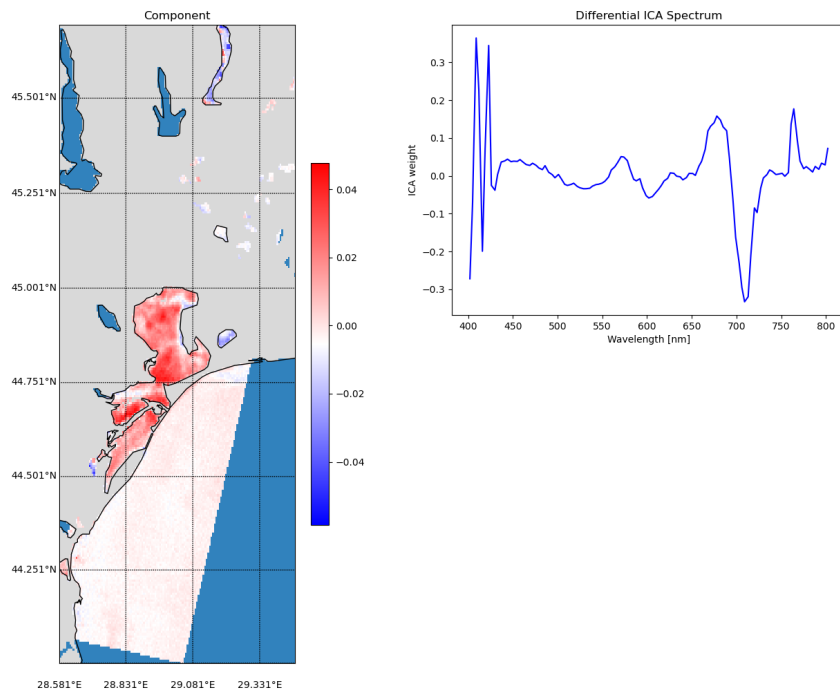


Figure 4.33: Spectral differencing component 3.

Spectral Difference Component 10 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

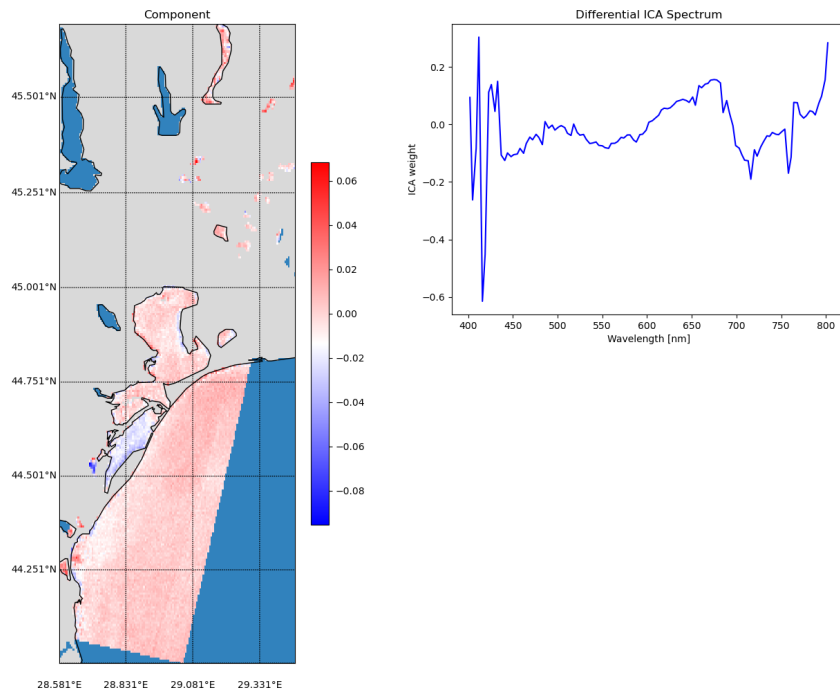


Figure 4.34: Spectral differencing component 10.

Spectral Differencing and Stacking Hybrid Results

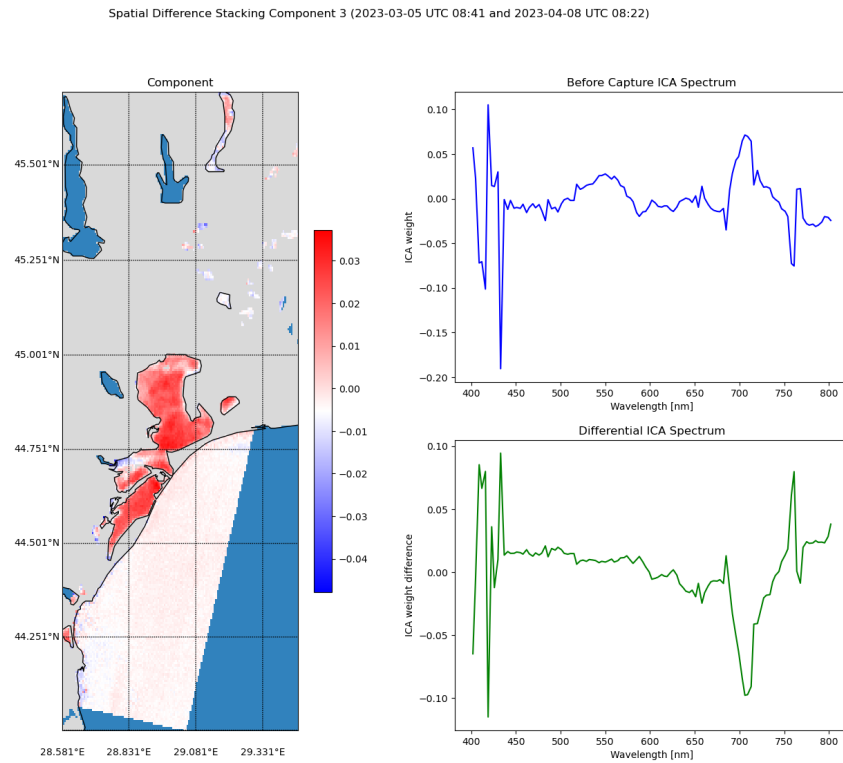


Figure 4.35: Spectral differencing and stacking component 3.

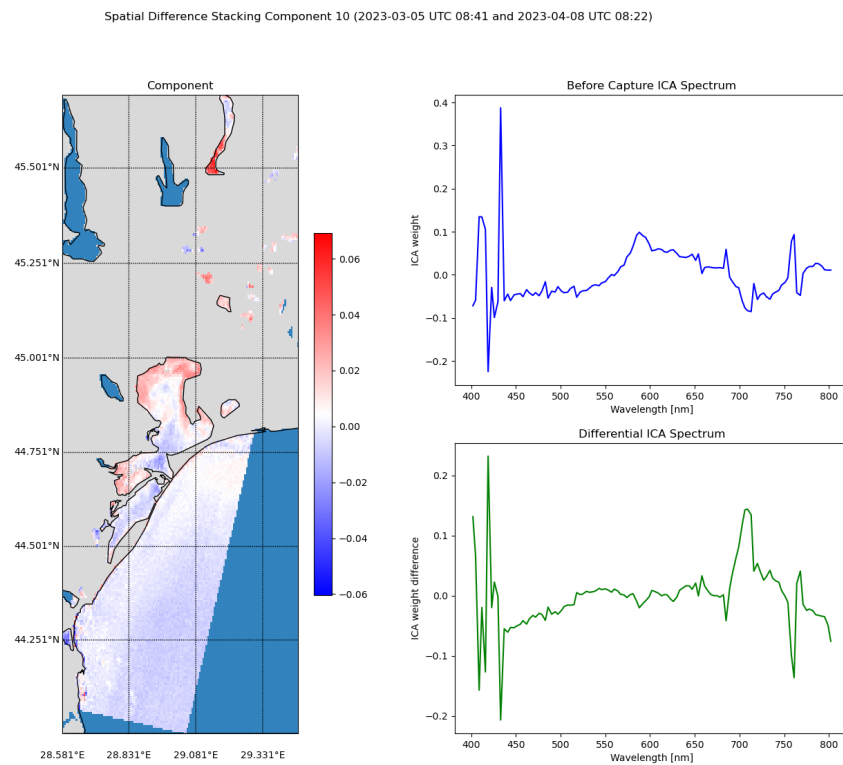


Figure 4.36: Spectral differencing and stacking component 10.

Component Matching Results

Random Un-mixing Matrix

Match: c7 and c10, Score: -0.8825601643212974 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

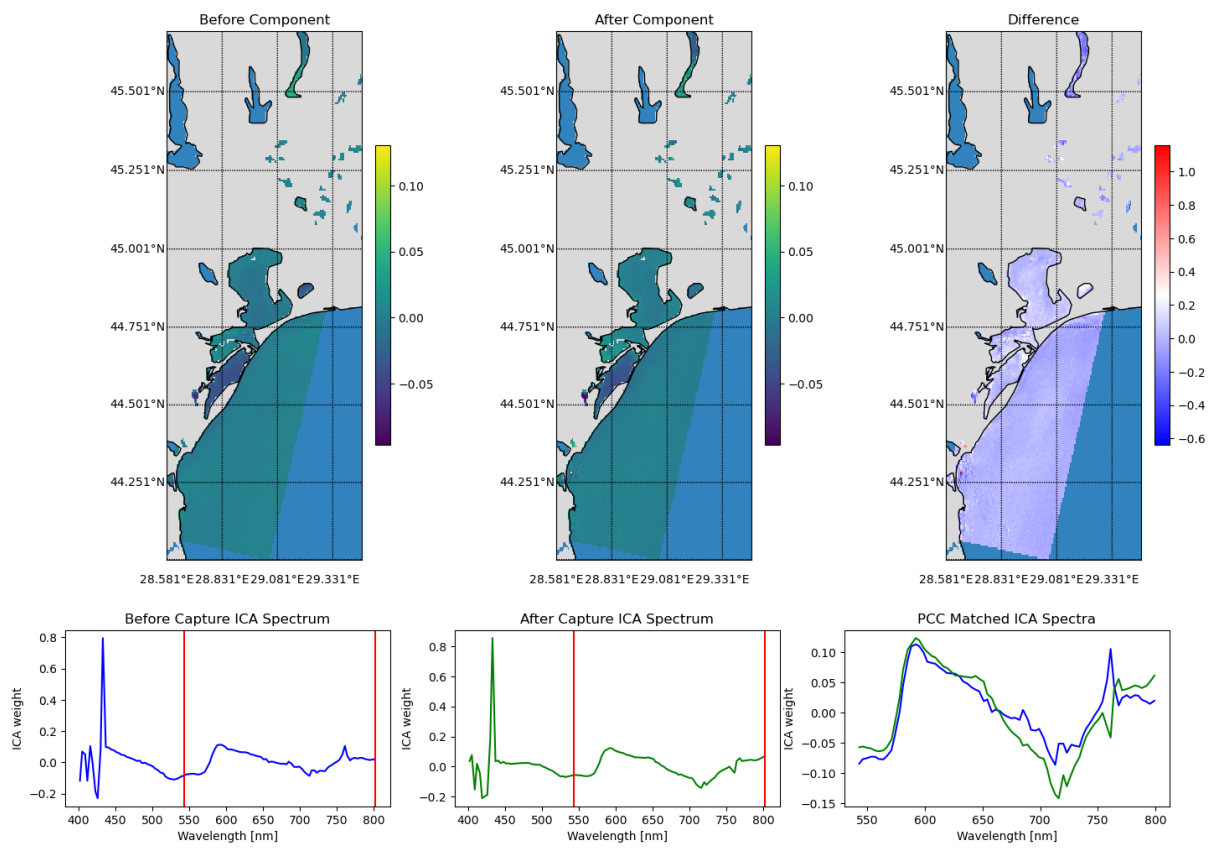


Figure 4.37: Component match 1.

Match: c6 and c8, Score: -0.7634384541500256 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

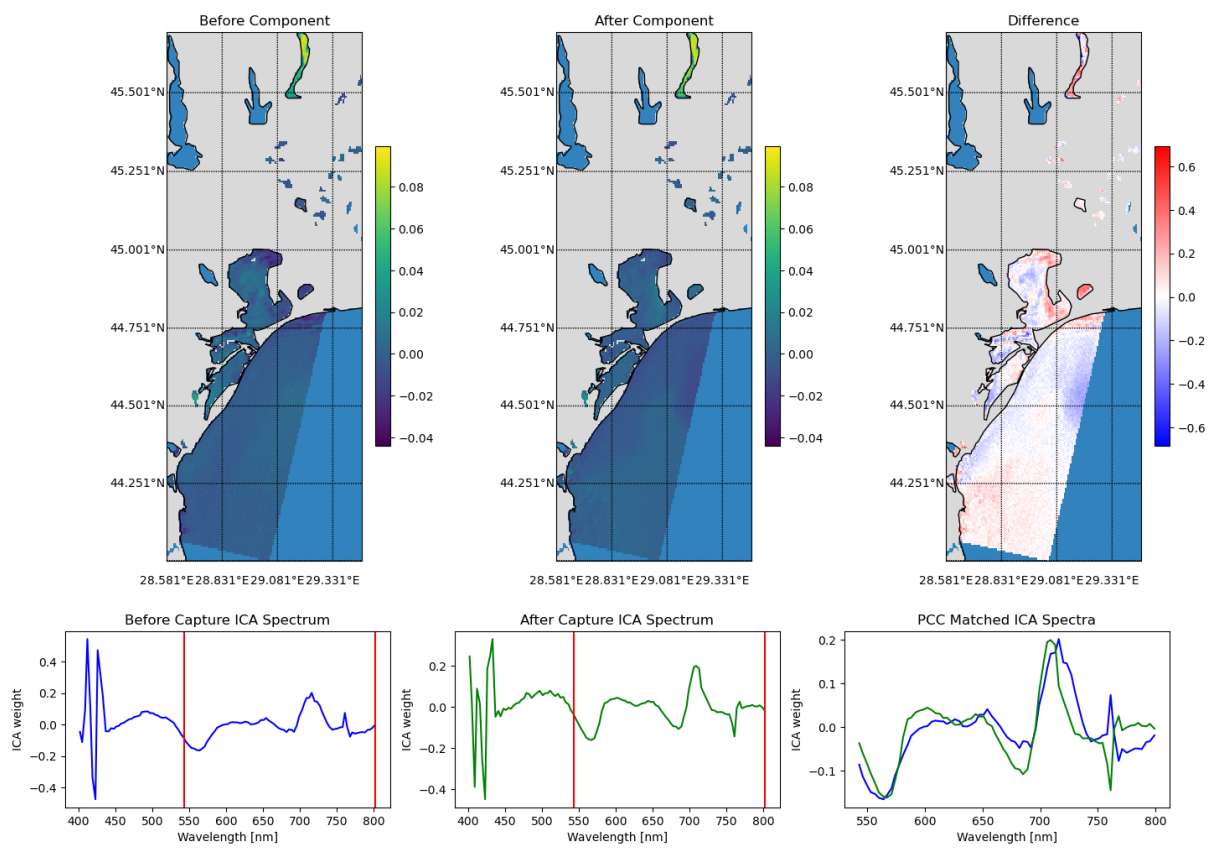


Figure 4.38: Component match 8.

Previous Un-mixing Matrix

Match: c7 and c4, Score: -0.9300681824876359 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

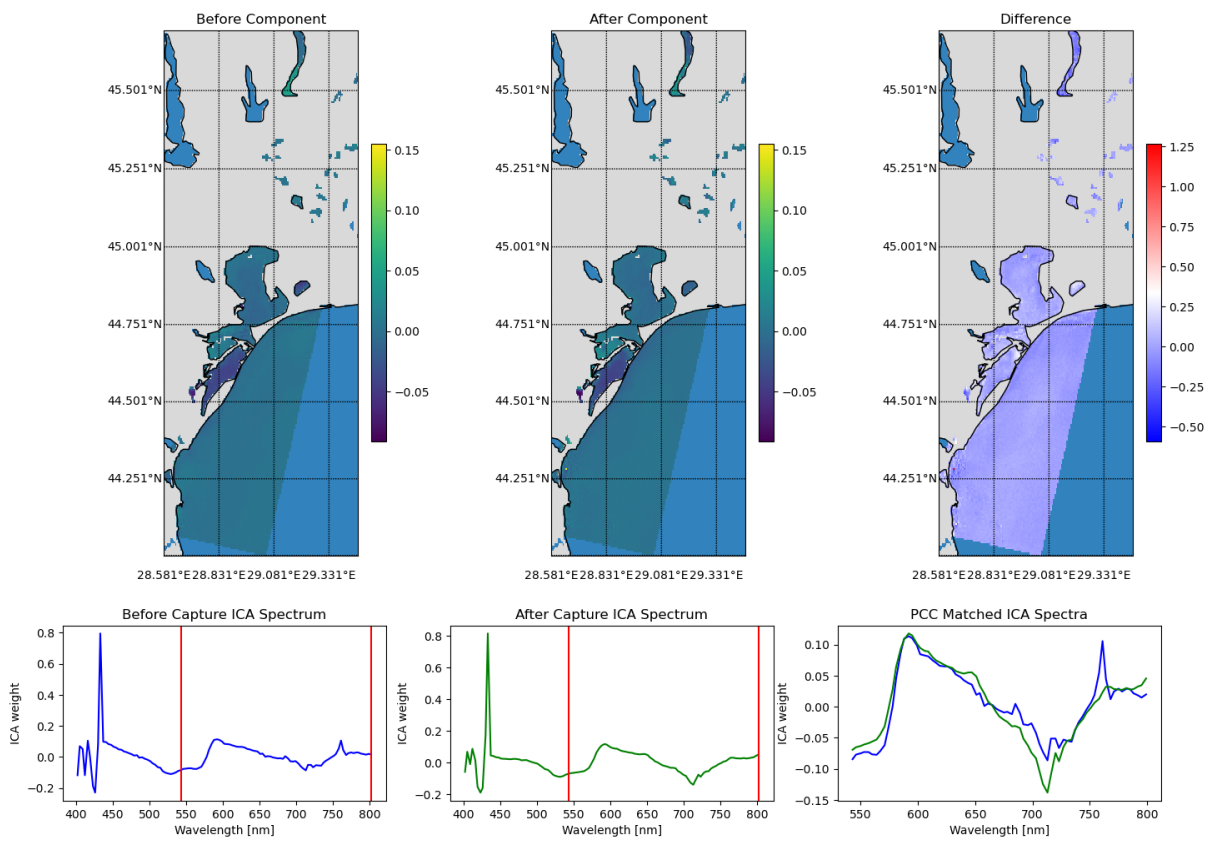


Figure 4.39: Component match 1.

Match: c6 and c8, Score: -0.8026448442637883 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

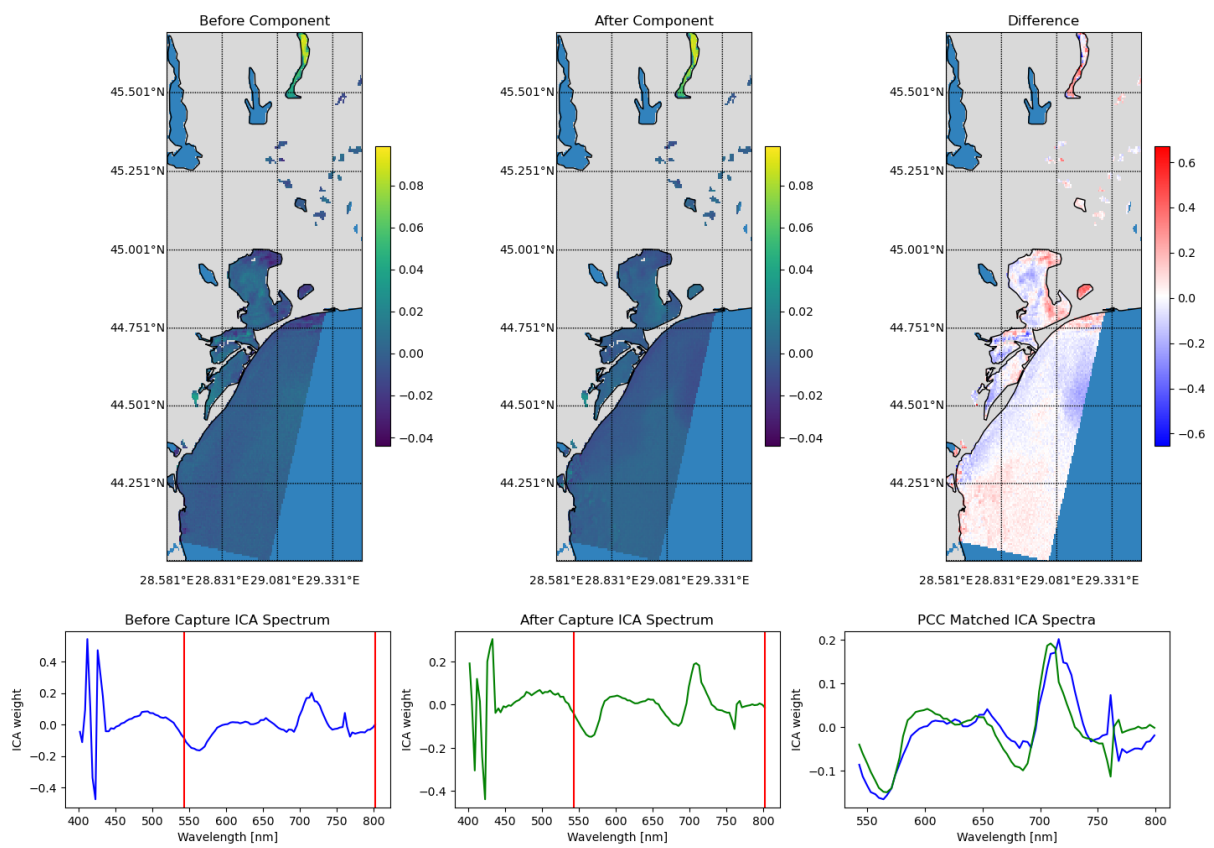


Figure 4.40: Component match 7.

ATGP-generated Un-mixing Matrix

Match: c7 and c8, Score: 0.917891942385279 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

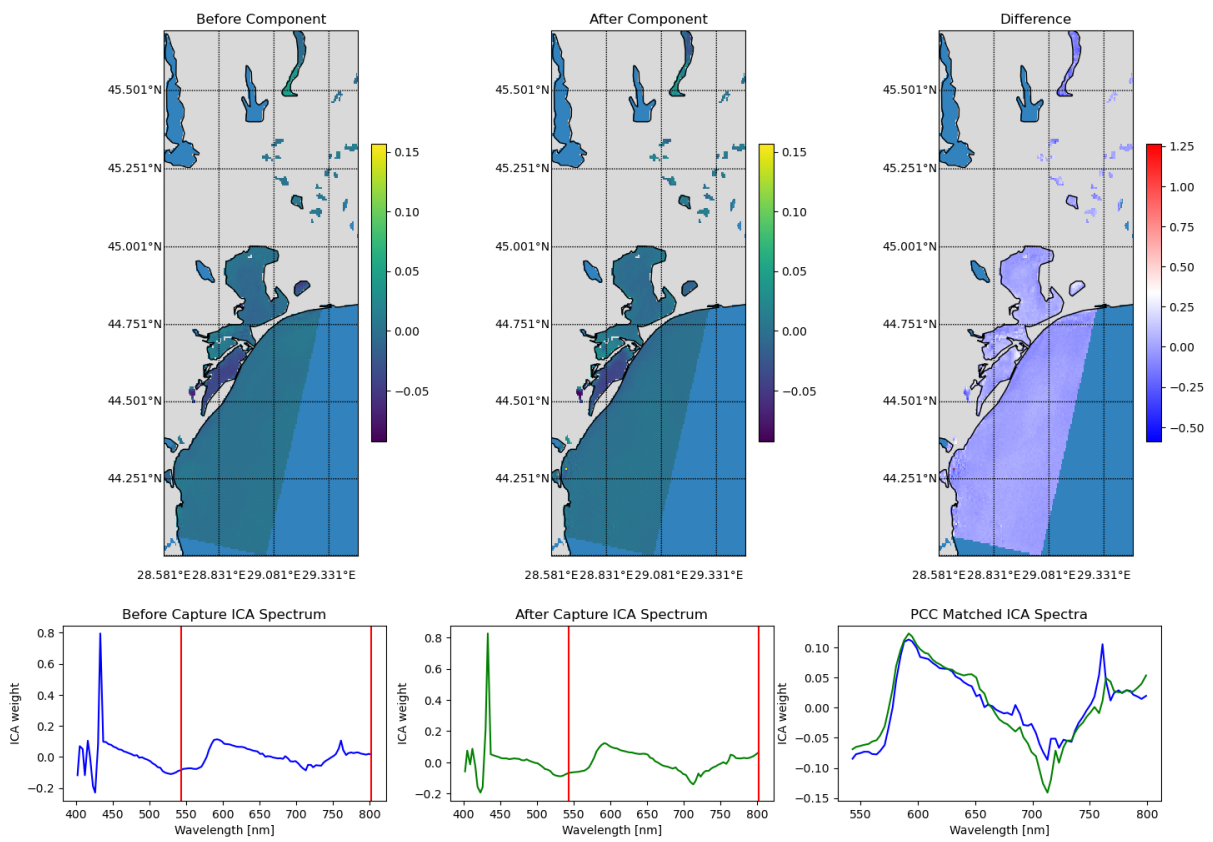


Figure 4.41: Component match 1.

Match: c6 and c7, Score: -0.8177958998101542 (2023-03-05 UTC 08:41 and 2023-04-08 UTC 08:22)

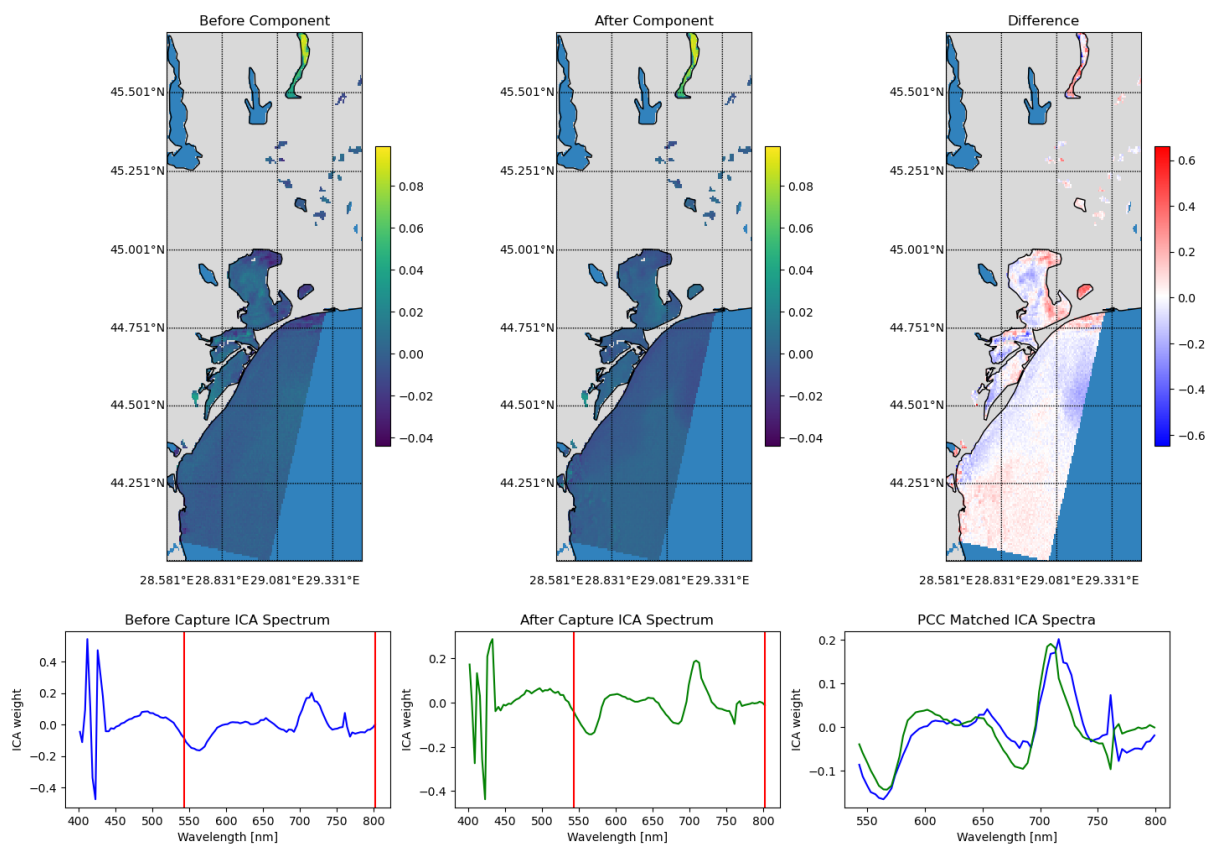


Figure 4.42: Component match 7.

4.6 Frohavet Change Detection Results

Description

The Frohavet change detection results are generated from two HYPSON-1 hyperspectral captures acquired on 28 March 2023 10:59 UTC and 29 March 2023 10:44 UTC. The target and HYPSON-1 data are discussed in Section 3.3.

Spatial Stacking Results

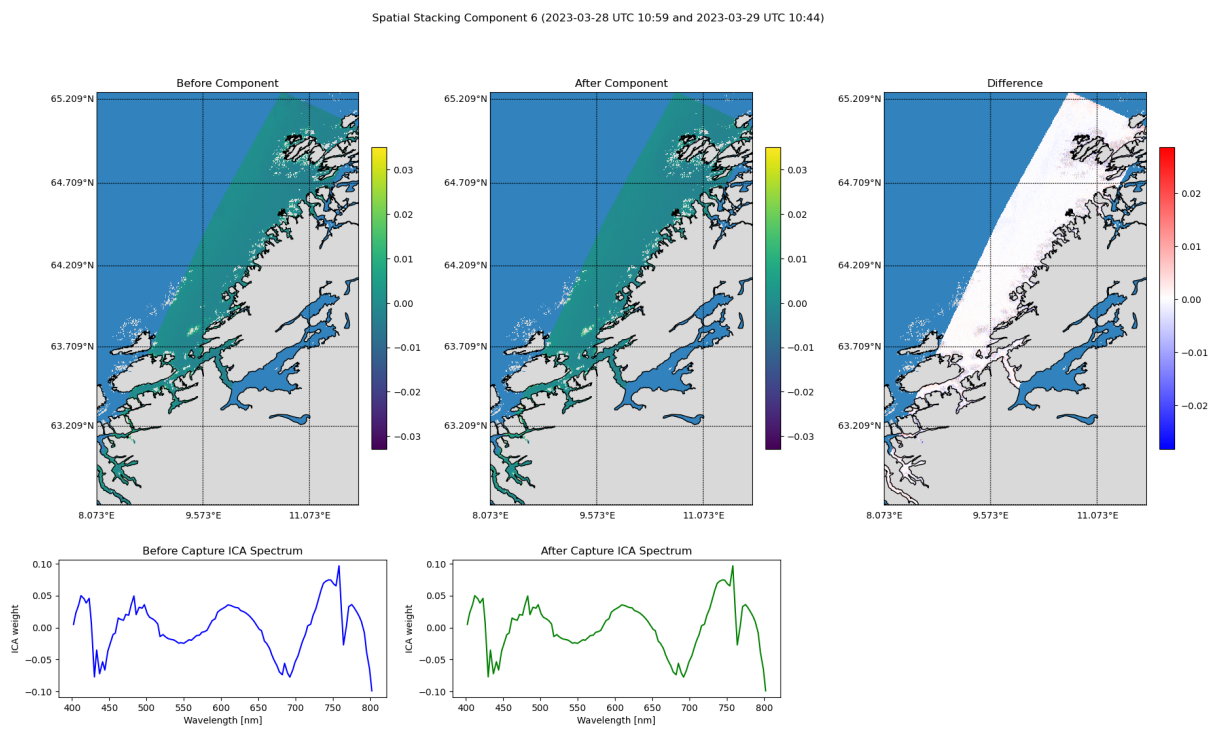


Figure 4.43: Spatial stacking component 6.

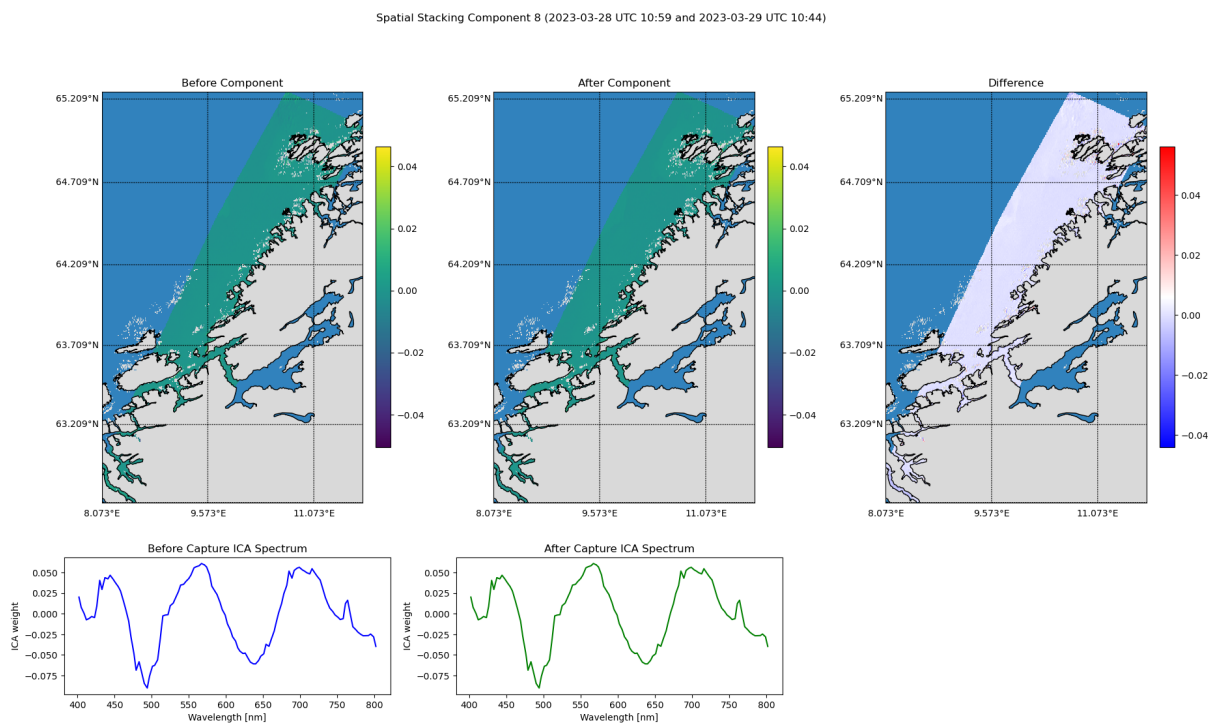


Figure 4.44: Spatial stacking component 8.

Spectral Stacking Results

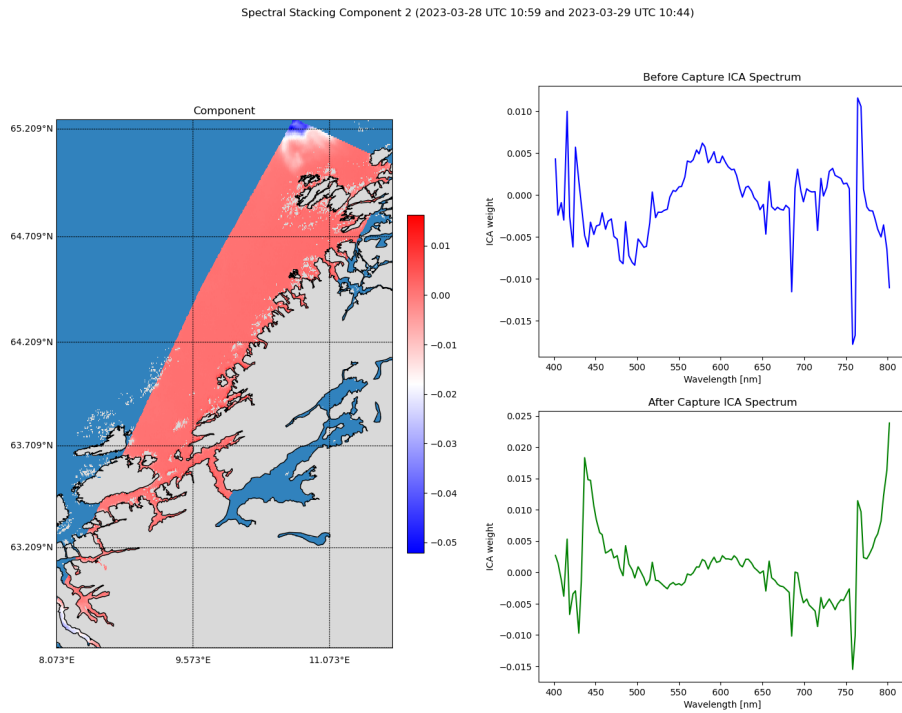


Figure 4.45: Spectral stacking component 2.

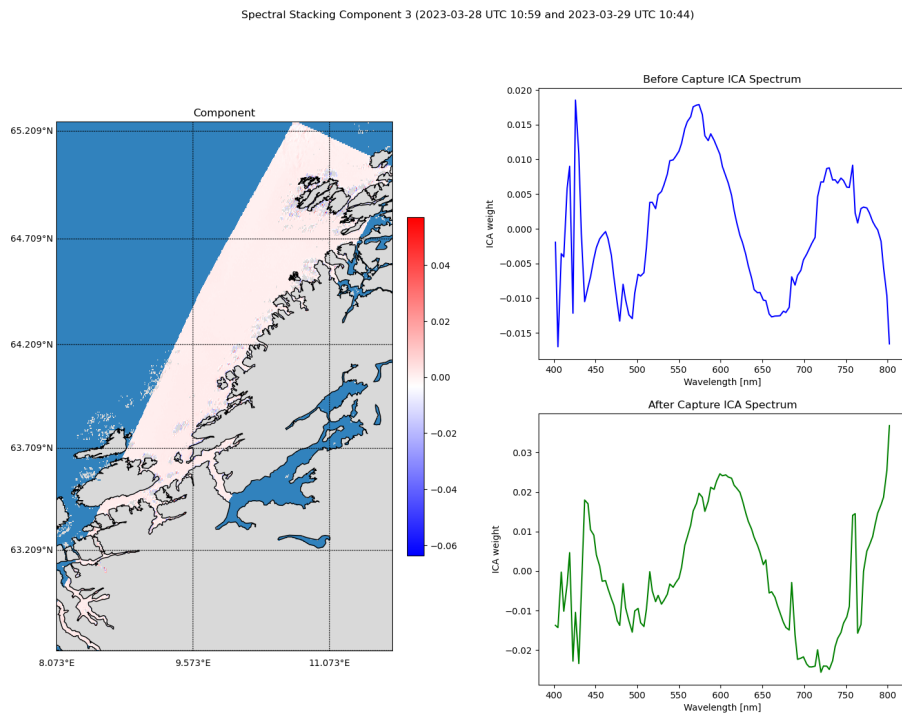


Figure 4.46: Spectral stacking component 3.

Spectral Differencing Results

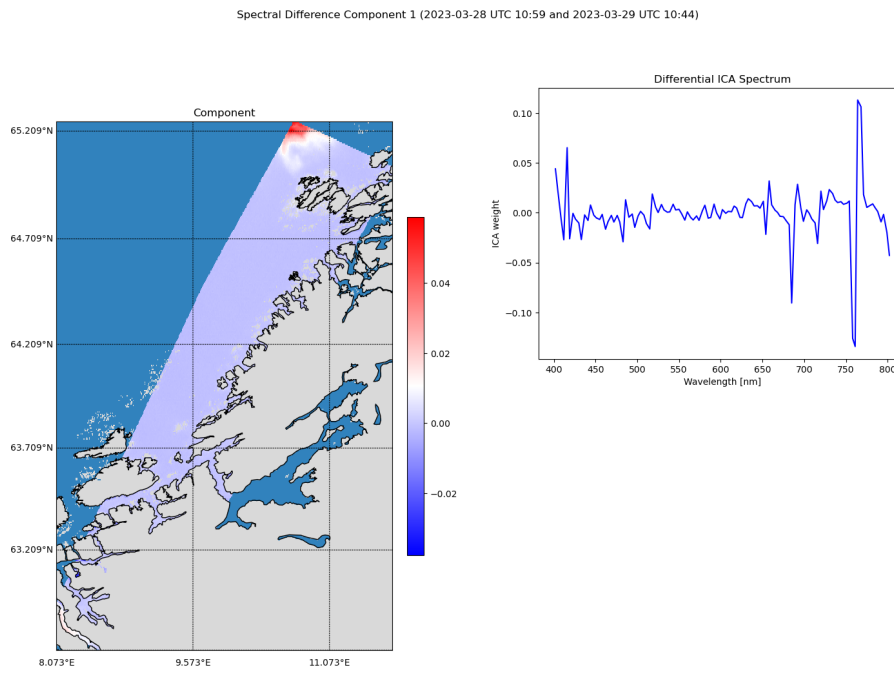


Figure 4.47: Spectral differencing component 1.

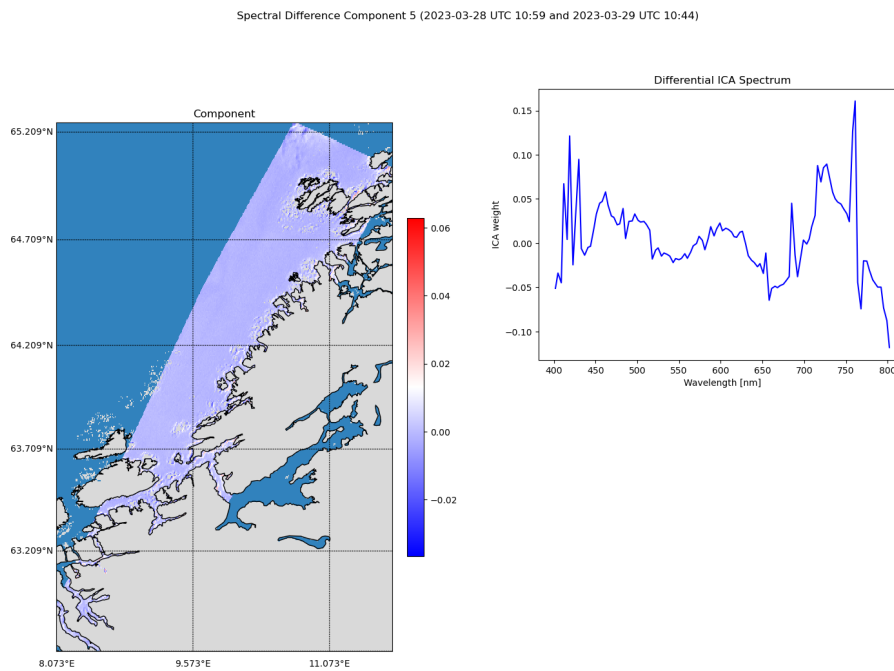


Figure 4.48: Spectral differencing component 5.

Spectral Differencing and Stacking Hybrid Results

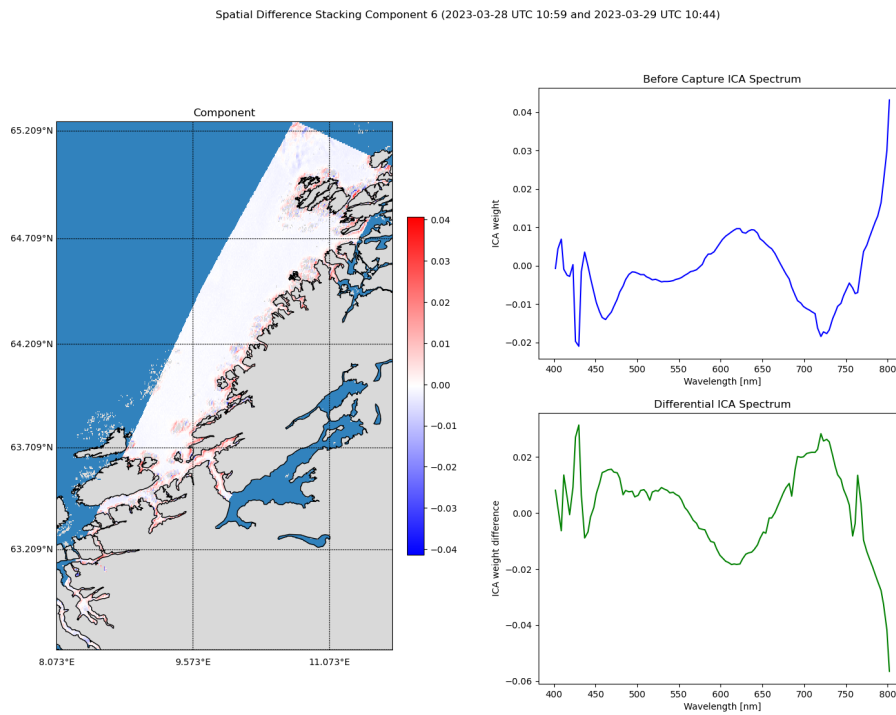


Figure 4.49: Spectral differencing and stacking component 6.

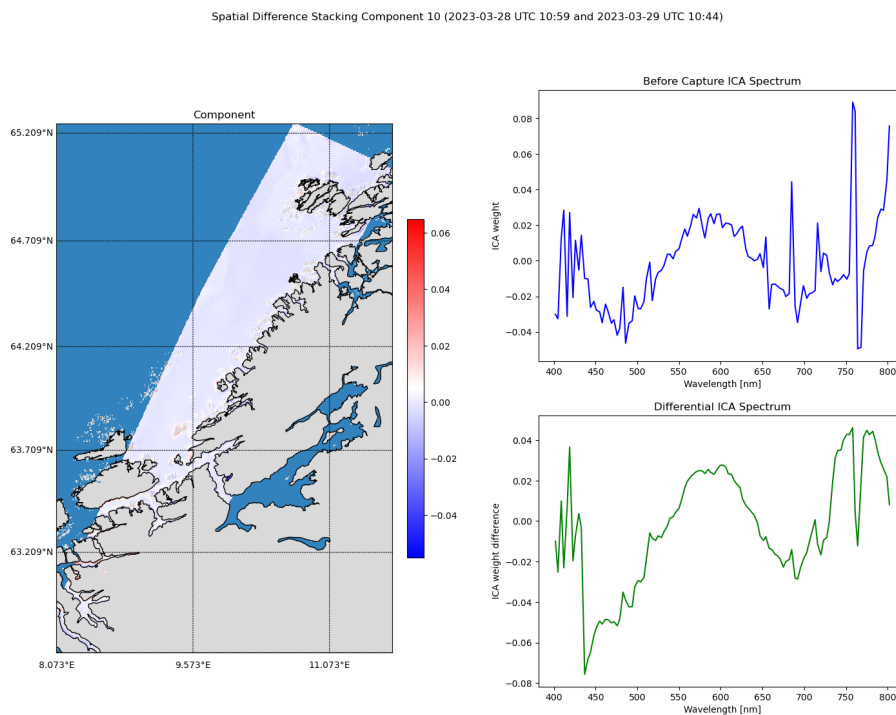


Figure 4.50: Spectral differencing and stacking component 10.

Component Matching Results

Random Un-mixing Matrix

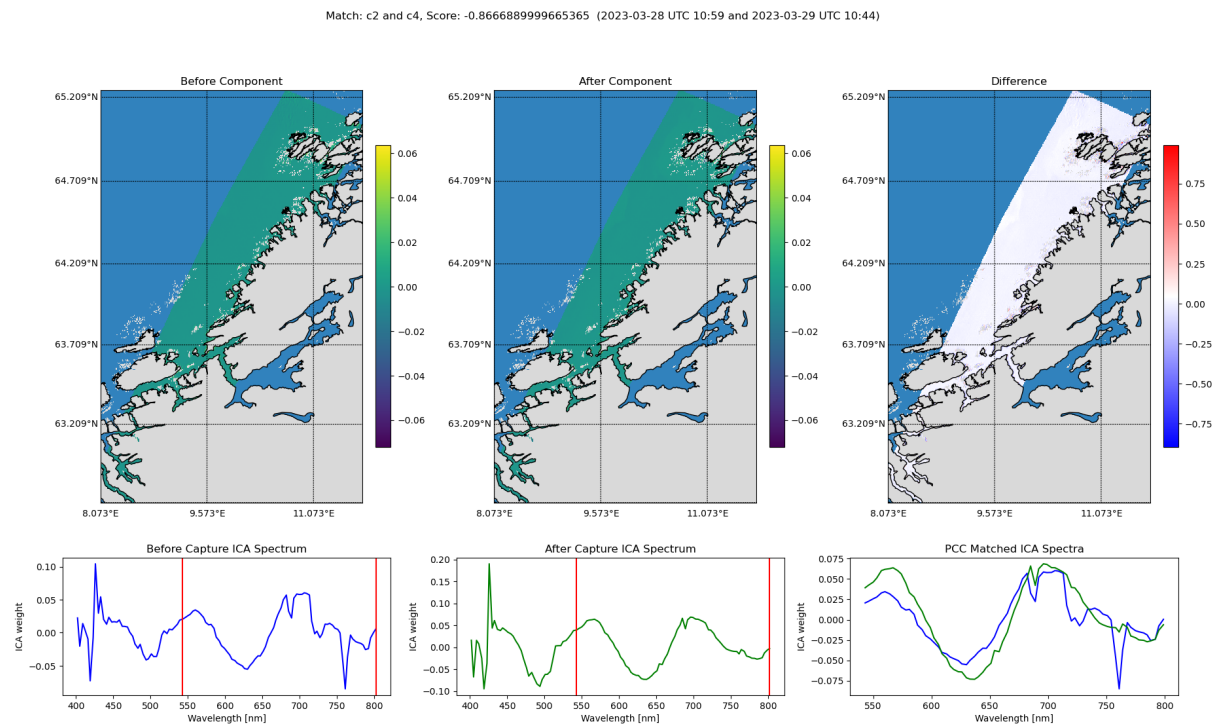


Figure 4.51: Component match 4.

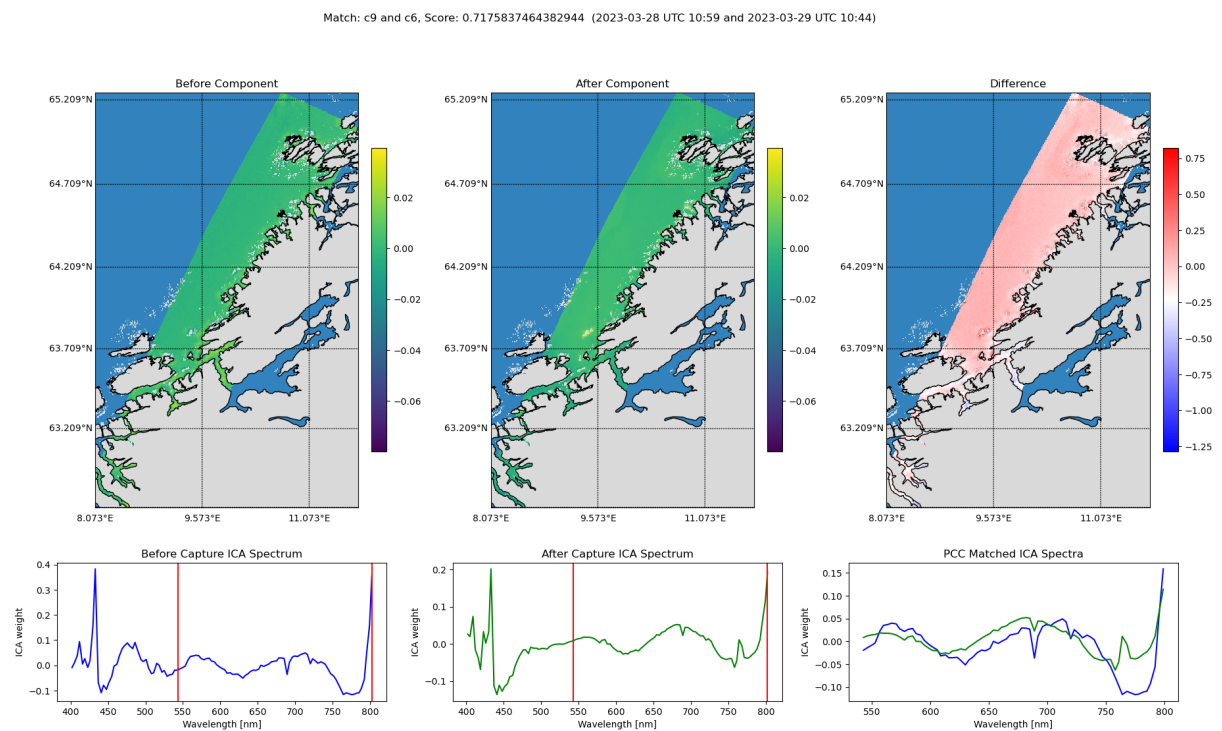


Figure 4.52: Component match 9.

Previous Un-mixing Matrix

Match: c2 and c3, Score: -0.9039813256722653 (2023-03-28 UTC 10:59 and 2023-03-29 UTC 10:44)

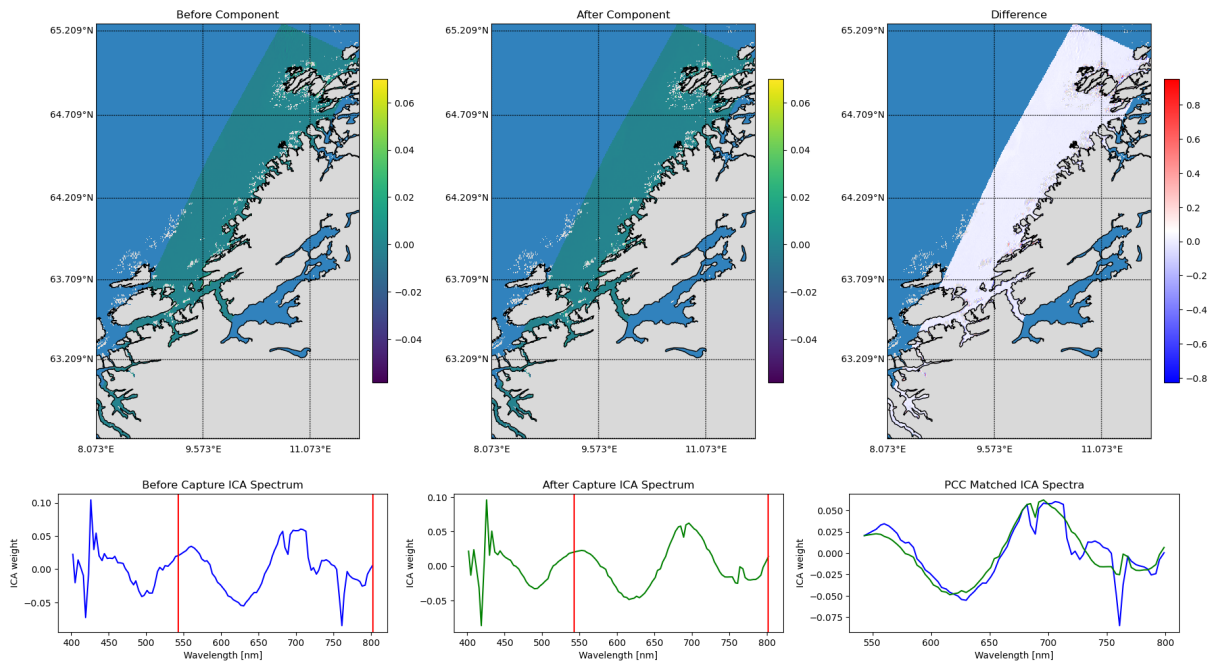


Figure 4.53: Component match 2.

Match: c9 and c8, Score: 0.8716809909918252 (2023-03-28 UTC 10:59 and 2023-03-29 UTC 10:44)

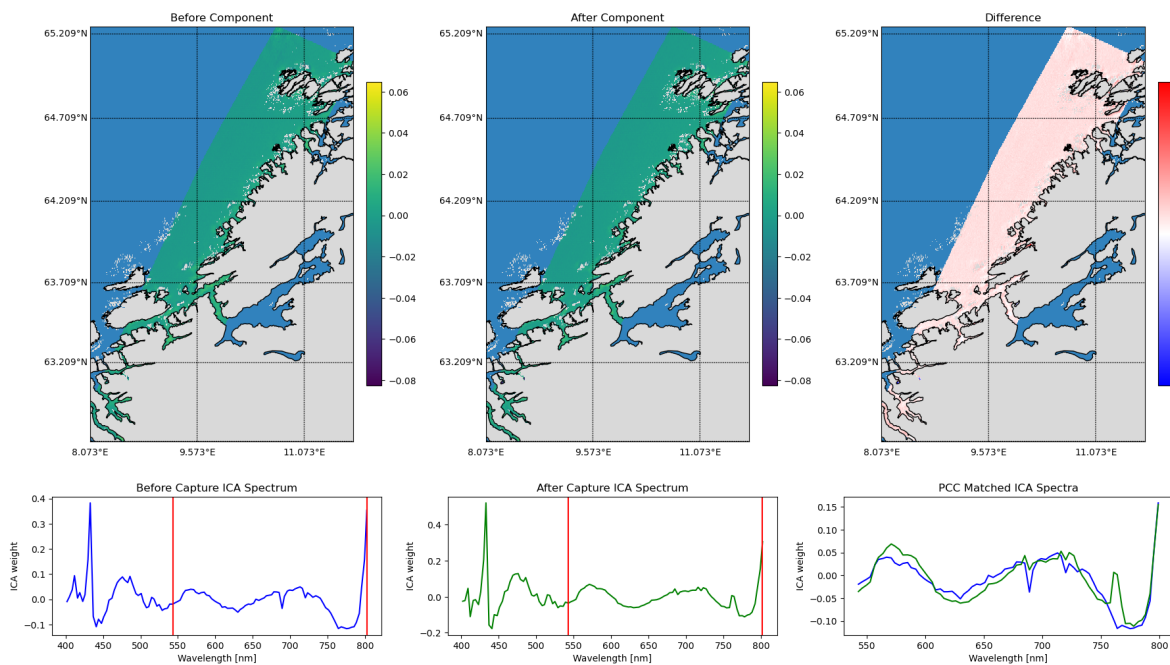


Figure 4.54: Component match 3.

ATGP-generated Un-mixing Matrix

Match: c8 and c10, Score: -0.8165777538299889 (2023-03-28 UTC 10:59 and 2023-03-29 UTC 10:44)

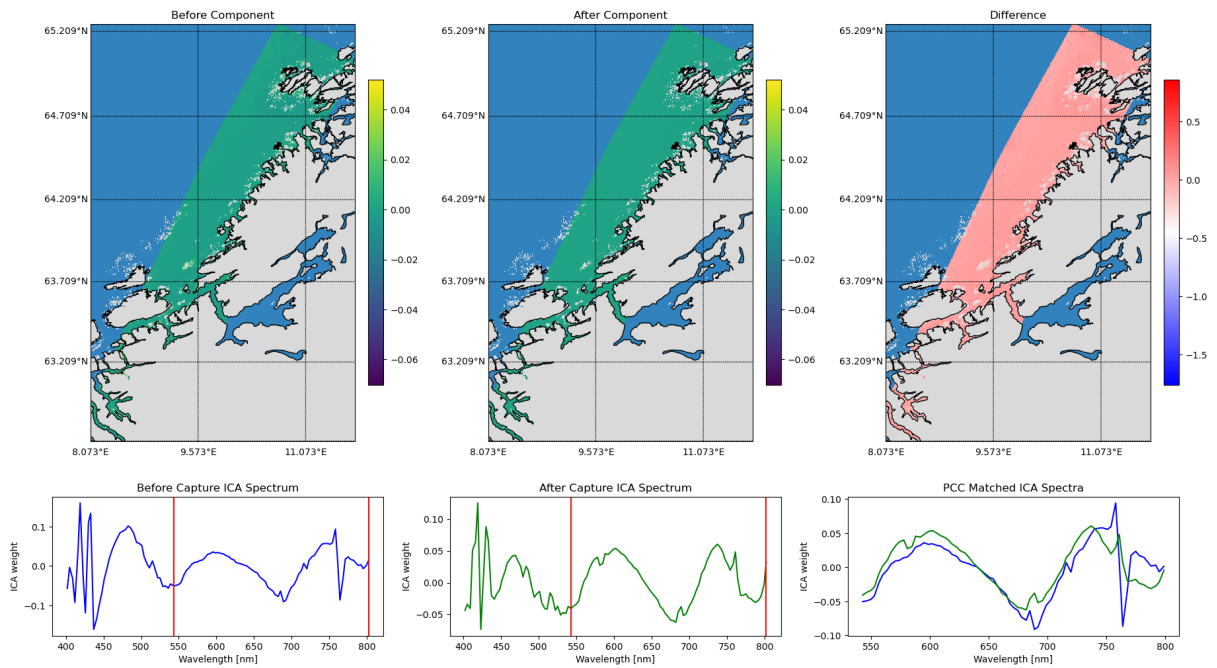


Figure 4.55: Component match 3.

Match: c2 and c3, Score: 0.8154711108956794 (2023-03-28 UTC 10:59 and 2023-03-29 UTC 10:44)

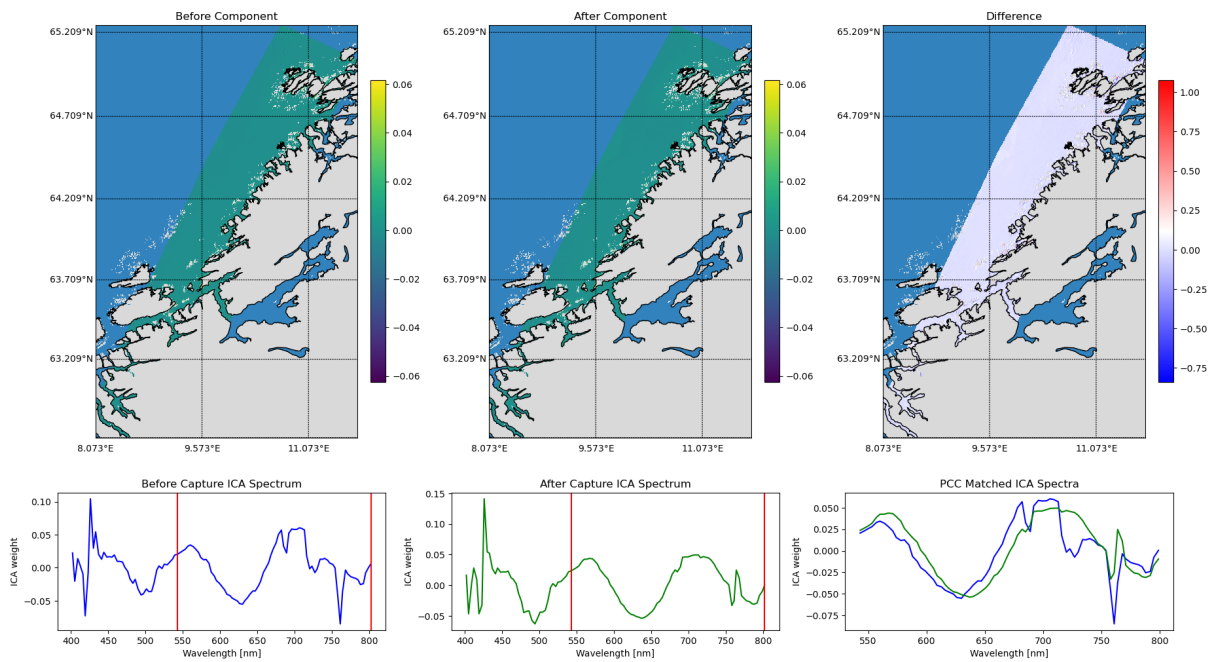


Figure 4.56: Component match 5.

Chapter 5

Discussion

5.1 Overview

This section primarily focuses on the interpretation of the HYPSON-1 change detection results and the comparison with validation data sources. Additionally, it evaluates the performance and capabilities of the ICA-based change detection techniques. Using unsupervised detection techniques like ICA on hyperspectral data poses challenges due to the inability to automatically identify the specific features represented by the independent components and change maps produced by ICA. Because ICA relies solely on data-driven analysis without the use of training data, there is no labeled truth data available for comparison with independent components or change maps. Consequently, many aspects of evaluating unsupervised ICA-based anomaly and change detection remain reliant on qualitative assessments.

5.2 Change Detection Techniques Comparison

One evaluation method for the ICA-based change detection techniques consists of directly comparing the HYPSON-1 change map results with change maps created from the validation data. Using this evaluation method, one can verify if there are any changing anomalies which correspond to algae or suspended matter based on spatial distribution shown in the change maps. Two sets of validation data were created from Sentinel-3 OLCI and NOAA HAB forecast sources and processed into change maps using Python or QGIS. The validation data sources are discussed in Section 3.7.

An important consideration is that both the Sentinel-3 OLCI and NOAA HAB forecast data products are not considered true ground truth data. Rather, they are estimates derived from multispectral observations, as opposed to algal concentrations or other physical quantities being directly measured in the ocean. However, the Sentinel-3 OLCI and NOAA data sources still serve as valuable references, since they were developed specifically to estimate algal and suspended matter concentrations. Using them allows verification of whether the spectral anomalies and changes detected in the HYPSON-1 data are also observed in the Sentinel-3 OLCI and NOAA data.

5.2.1 Comparison to Sentinel-3 OLCI Validation Data

The Sentinel-3 OLCI data serves as a validation dataset for all four sites due to the global coverage offered by the Sentinel-3A and Sentinel-3B satellite. The validation data has two variable estimates:

algal concentration (Chl-a) and total suspended matter (TSM). The validation data was processed into plots depicting the Chl-a and TSM estimates from the first and second Sentinel-3 images as well as a change map plot. The change map plot represented the direct difference in values between the first and second images and can be used for comparison with the HYPSON-1 change maps. Within the Sentinel-3 validation dataset, the Chl-a and TSM concentration estimates were both generated using the Inverse Modelling Technique neural network discussed in Section 3.7 rather than the OC4Me algorithm. This was not a choice, rather they were the only variables available for the Sentinel-3 observations coinciding with the HYPSON-1 captures.

The Chl-a and TSM concentration products from Sentinel-3 are measured in units of mass per volume. Chl-a is measured in mgm^{-3} while TSM is measured in gm^{-3} . Changes are measured in Δmgm^{-3} and Δgm^{-3} for Chl-a and TSM, respectively. The following subsections compare the Sentinel-3 validation data change maps with HYPSON-1 change detection results for each of the observational targets.

Lake Erie

The Sentinel-3 OLCI Chl-a and TSM estimates for Lake Erie are shown in Figure 5.2.2 and 5.2, respectively. The Sentinel-3 observations were made within 15 minutes of the HYPSON-1 captures, as shown in Table 5.4.

Table 5.1: Image acquisition times comparison for Lake Erie.

HYPSON-1	Sentinel-3	Difference
19 July 2022 15:50 UTC	19 July 2022 16:03 UTC	13 minutes
27 August 2022 16:05 UTC	27 August 2022 15:51 UTC	-14 minutes

The Lake Erie results are presented in Section 4.3. In the Sentinel-3 change maps, it can be seen that spatial distributions of both the Chl-a and TSM estimates exhibit a close resemblance, indicating that algae is likely the primary contributor to the suspended matter concentrations observed in the Sentinel-3 data for Lake Erie. Similar spatial distributions are observed in several of the Lake Erie change maps in Section 4.3. Notable results include:

- Spatial stacking component 7 (Figure 4.2)
- Spectral stacking component 7 (Figure 4.4)
- Spectral differencing component 4 (Figure 4.5)
- Spectral differencing and stacking component 5 (Figure 4.7)

All four of these change maps show a distinct pattern of a diminishing anomaly in the lower left of the images (keeping in mind that some of the components may have their sign reversed). This corresponds to the diminishing concentration within the same area in the Sentinel-3 change maps. The same changing anomaly is detected in the component matching techniques. The matches containing the anomaly include:

- Random un-mixing matrix init. match 3 (Figure 4.9)
- Previous un-mixing matrix init. match 3 (Figure 4.11)

- ATGP-generated un-mixing matrix init. match 4 (Figure 4.14)

In terms of the relative reflectance spectrum, the anomaly is harder to interpret. The component matching result in Figure 4.14 provides the one of clearest spectrum examples from the Lake Erie results to analyze. The before and after components have local peaks around 700 nm. This peak could be associated with cyanobacteria that is known to cause HABs at that location in Lake Erie. However, there should also be a peak at 450 nm which is not present in the components extracted from the captures. The spatial distribution of the change provides strong evidence that the changing anomaly is algal but it cannot be confirmed without full understanding the spectra recovered from the HYPSON-1 results.

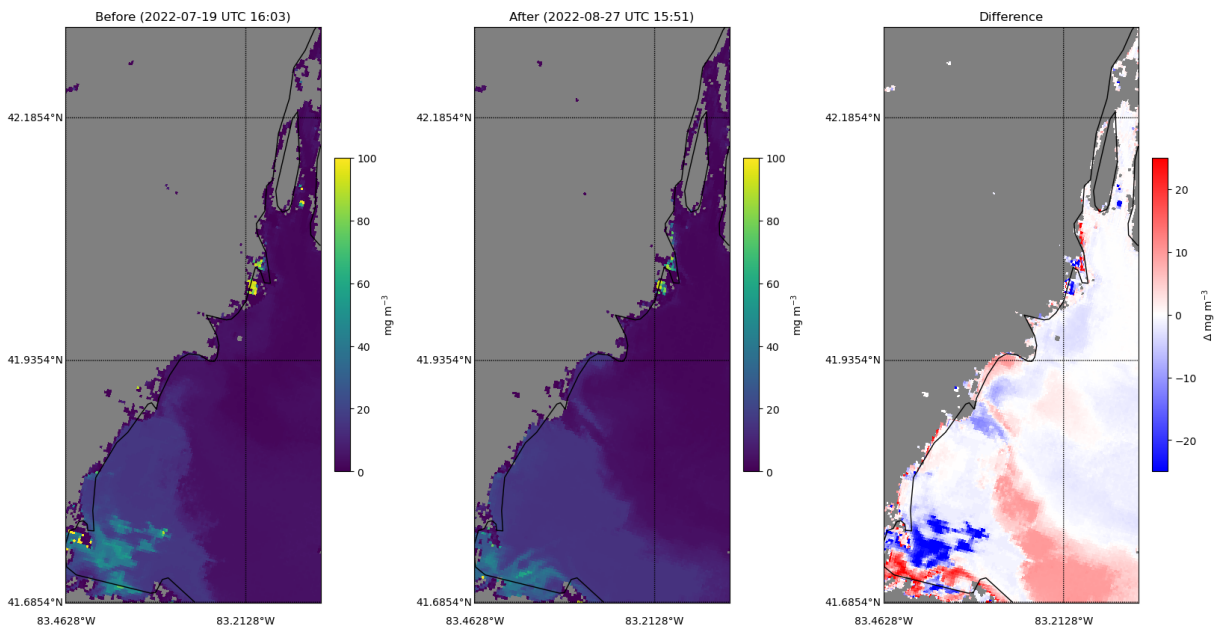


Figure 5.1: Sentinel-3 Chl-a estimates for Lake Erie.

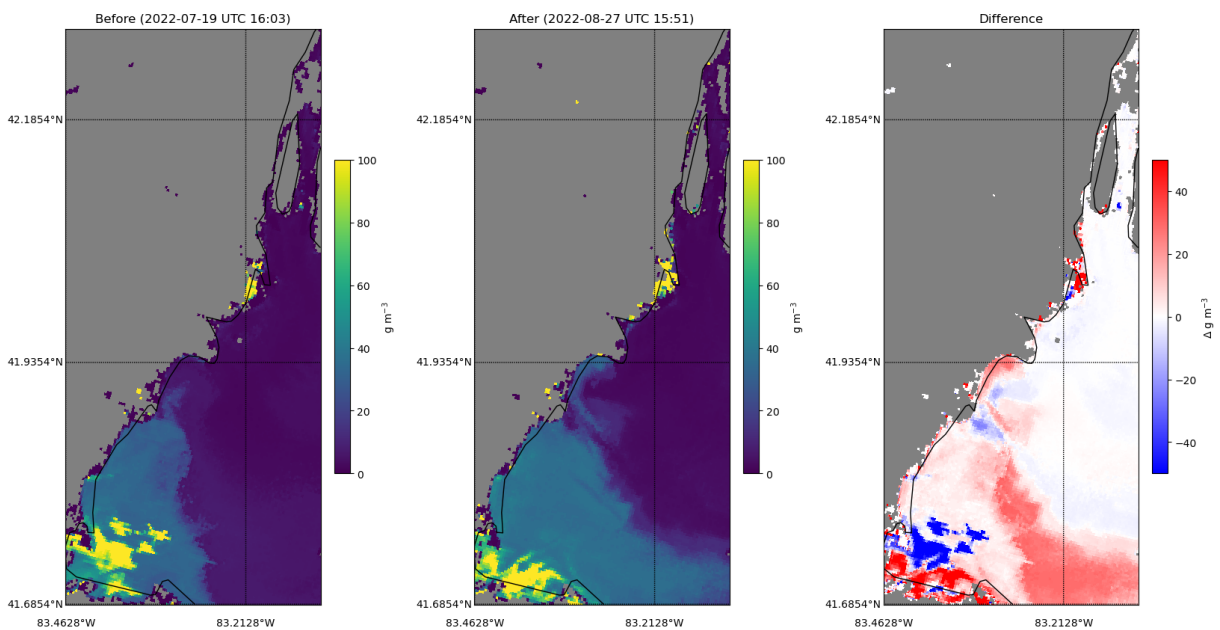


Figure 5.2: Sentinel-3 TSM estimates for Lake Erie.

Salish Sea

The Sentinel-3 OLCI Chl-a and TSM estimates for the Salish Sea are shown in Figure 5.3 and 5.4, respectively. The Sentinel-3 observations were made within 20 minutes of the HYPSON-1 captures, as shown in Table 5.2.

Table 5.2: Image acquisition times comparison for the Salish Sea.

HYPSON-1	Sentinel-3	Difference
12 July 2022 18:46 UTC	12 July 2022 19:06 UTC	20 minutes
13 July 2022 18:34 UTC	13 July 2022 18:40 UTC	6 minutes

The Salish Sea results are presented in Section 4.4. As was seen in the Lake Erie Sentinel-3 validation data, both the Chl-a and TSM estimates correspond to each other, again indicating that algae is likely the primary contributor to the suspended matter concentrations in the Salish Sea. Several of the HYPSON-1 Salish Sea change maps from Section 4.4 show spatial patterns similar to the Sentinel-3 Chl-a change maps:

- Spatial stacking component 3 (Figure 4.15)
- Spectral stacking component 4 (Figure 4.18)
- Spectral differencing and stacking component 5 (Figure 4.21)
- ATGP-generated un-mixing matrix init. match 4 (Figure 4.28)

Of these, the ATGP-generated un-mixing matrix match 4 result (Figure 4.28) is the most interesting. Its change maps contains a spatial pattern in the upper right similar to the Sentinel-3 Chl-a change map. Based solely on this though, it is not possible to conclude that the detected change is due to algae since there are no distinct spectral peak in the spectrum that are associated with chlorophyll.

Another interesting result from the Salish Sea results is that the spectral differencing component 2 (Figure 4.19) change map resembles the Sentinel-3 TSM change map closer than the Sentinel-3 Chl-a change map. This might be a result of suspended matter or some other feature being detected however because the spectral differencing component spectrum only shows the change in the relative reflectance values it is not possible to identify the feature involved in this change.

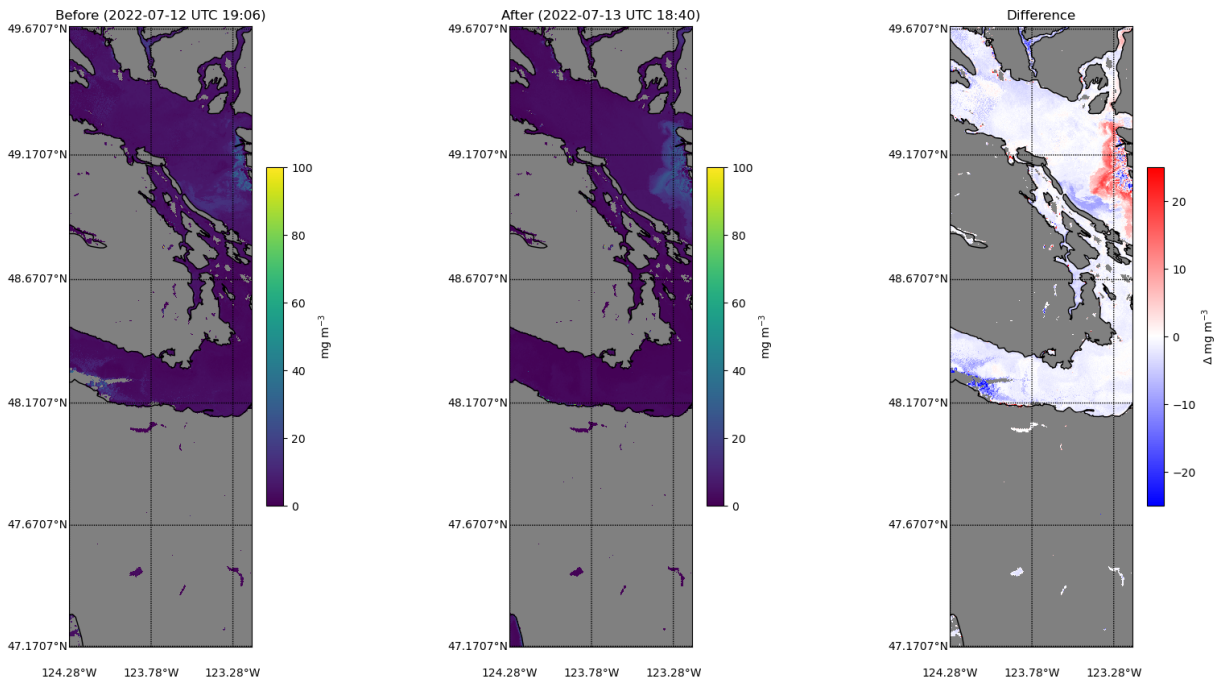


Figure 5.3: Sentinel-3 Chl-a estimates for the Salish Sea.

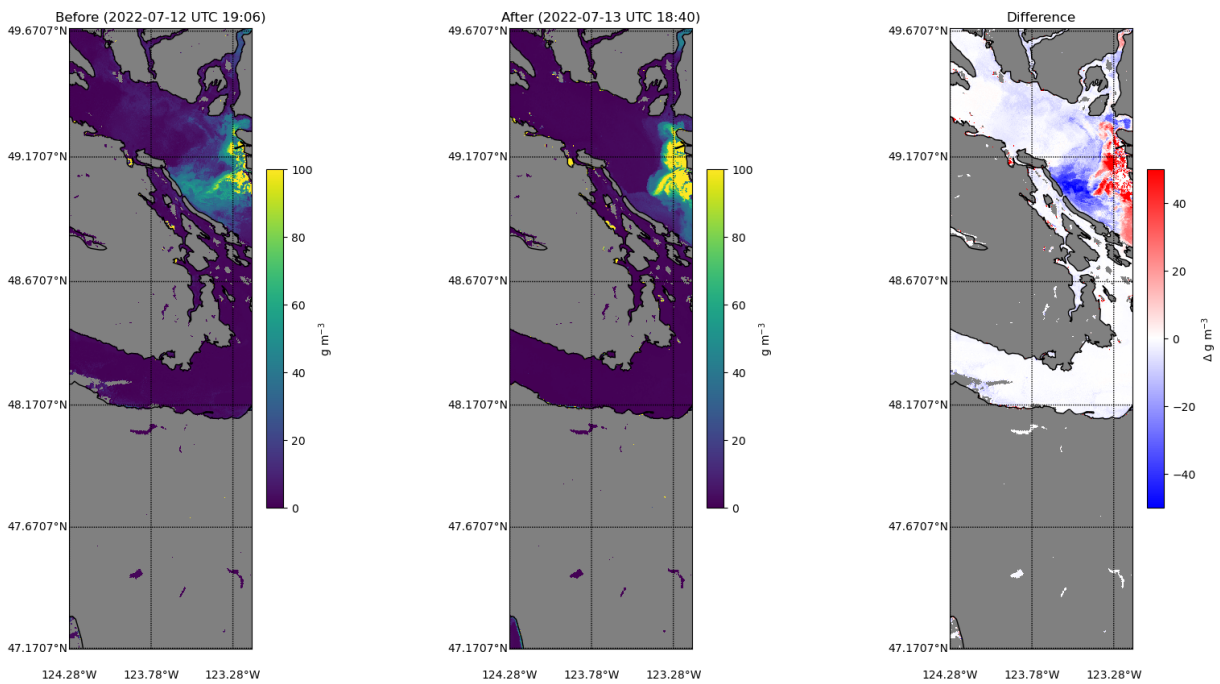


Figure 5.4: Sentinel-3 TSM estimates for the Salish Sea.

Danube River Delta

The Sentinel-3 OLCI Chl-a and TSM estimates for the Danube River Delta are shown in Figure 5.5 and 5.6, respectively. The Sentinel-3 observations were made within 25 minutes of the HYPSON-1 captures, as shown in Table 5.3. The Sentinel-3 observation of the Danube River Delta made on 05 March 2022 was captured at exactly the same time as the HYPSON-1 capture, making that particular date an interesting point of comparison between the two data sources.

Table 5.3: Image acquisition times comparison for the Danube River Delta.

HYPISO-1	Sentinel-3	Difference
05 March 2023 08:41 UTC	05 March 2023 08:41 UTC	0 minutes
08 April 2023 08:22 UTC	08 April 2023 07:58 UTC	-24 minutes

The Danube River Delta results are presented in Section 4.5. The Danube River Delta target results are interesting to examine because of the significant differences between the various lakes at the site. A map of the region is shown in Figure 3.8.

For the Danube River Delta target, the Sentinel-3 Chl-a and TSM estimates exhibit larger differences from each other than the Chl-a and TSM estimates in the Lake Erie and Salish Sea sites. In particular, the Sentinel-3 TSM estimates show that Lake Sinoe has a high concentration of suspended matter in the water compared to the other surrounding lakes. Lake Sinoe is not connected to the other lakes so it is possible the suspended matter is caused by non-algal sources such as sediments or run-off from agriculture or streams flowing into Lake Sinoe. The HYPISO-1 change maps that exhibit similarities to the Sentinel-3 TSM map include:

- Spatial stacking component 7 (Figure 4.29)
- Spectral stacking components 6 and 9 (Figure 4.31 and Figure 4.32)
- Spectral differencing component 10 (Figure 4.34)

Of these results, the spectral stacking components 6 (Figure 4.31) is the best example, showing a close pattern to the Sentinel-3 TSM change map as well as both an before capture spectrum and an after capture spectrum. When the signs of the spectra are reversed, the spectra closely resemble the spectra matched in the previous un-mixing matrix initialization match 7 (Figure 4.42) and ATGP-generated un-mixing matrix initialization match 7 (Figure 4.42) results. These three different techniques each detect spectral features corresponding to the suspended matter concentration change in Lake Sinoe.

The HYPISO-1 change detection results also produced change maps with correspondence to the Sentinel-3 algal concentration change map. The Sentinel-3 Chl-a estimates shows that there is an increase of algal concentration on the northern and eastern shores of Lake Razim. The change detection results that reflect the increase in algal concentration are:

- Spatial stacking component 7 and 8 (Figure 4.29 and Figure 4.30)
- Spectral differencing component 3 (Figure 4.33)
- Spectral differencing and stacking component 10 (Figure 4.36)

The component matching techniques also had change maps featuring similar spatial distributions as the increase in algal concentration in the Sentinel-3 data:

- Random un-mixing matrix init. match 8 (Figure 4.38)
- Previous un-mixing matrix init. match 7 (Figure 4.40)
- ATGP-generated un-mixing matrix init. match 7 (Figure 4.42)

The results from the component matching techniques all show similar spectra in the before and after components. The spectra have local spectral peaks near 700 nm, a feature that is associated with chlorophyll. However, with no peak around 500 nm, it is difficult to definitively attribute the spectra and change maps to algae.

One difficulty with analyzing the presence of a cloud formation running through Lake Golovita in the 05 March 2023 Sentinel-3 and HYPSON-1 captures. Spectral signals from the cloud formation appear in many of the Danube River Delta components and change maps. A clear example of interference from clouds can be seen in spectral stacking component 9 (Figure 4.32). These results demonstrate a situation where a cloud mask would improve the performance of ICA-based anomaly and change detection by removing undesired cloud features.

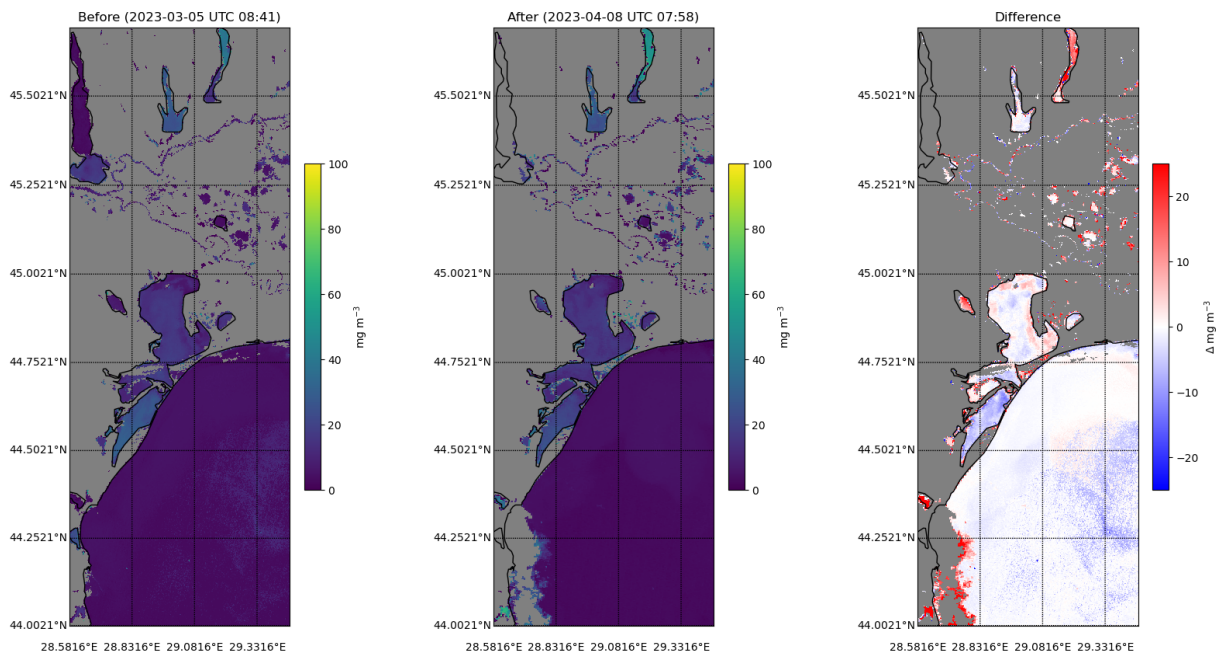


Figure 5.5: Sentinel-3 Chl-a estimates for the Danube River Delta.

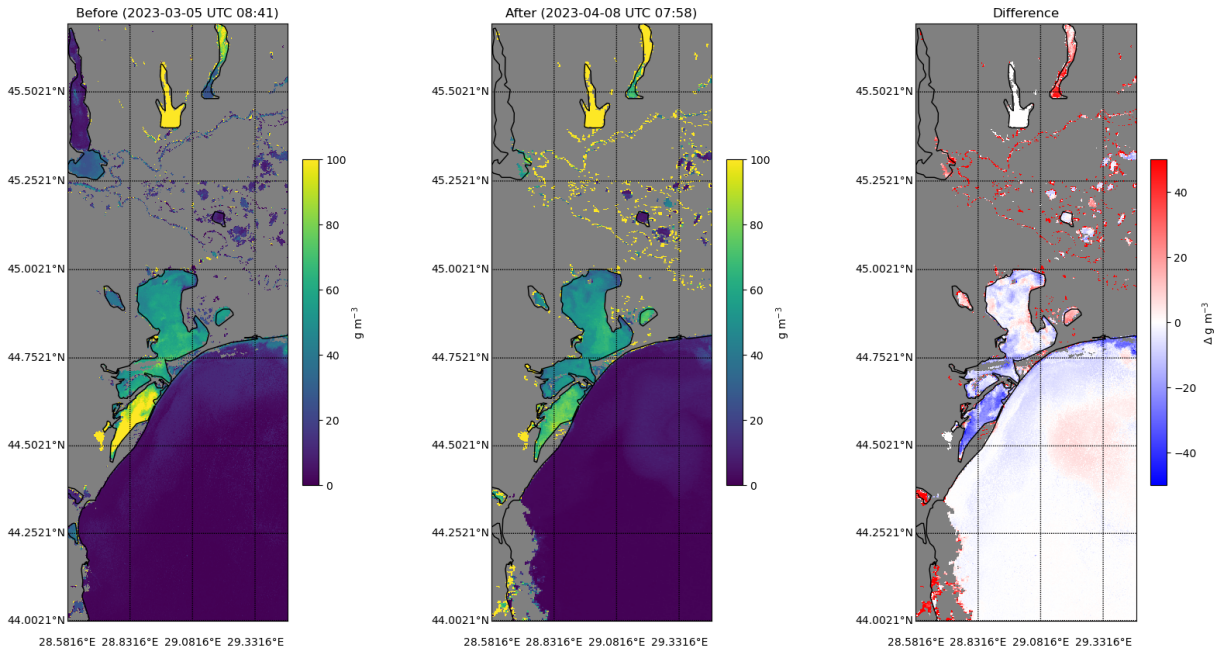


Figure 5.6: Sentinel-3 TSM estimates for the Danube River Delta.

Frohavet

The Sentinel-3 OLCI Chl-a and TSM estimates for Frohavet are shown in Figure 5.2.2 and 5.2, respectively. The Sentinel-3 observations were made within 15 minutes of the HYPISO-1 captures, as shown in Table 5.4.

Table 5.4: Image acquisition times comparison for Frohavet.

HYPISO-1	Sentinel-3	Difference
28 March 2023 10:59 UTC	28 March 2023 10:20 UTC	-39 minutes
29 March 2023 10:44 UTC	29 March 2023 09:54 UTC	-50 minutes

The Frohavet results are presented in Section 4.6. The Frohavet target was the least active site of the four targets in terms of Chl-a and TSM estimates from Sentinel-3. The Sentinel-3 data shows that was chlorophyll and suspended matter in the ocean outside the coverage of the HYPISO-1 captures. The data also shows that there was chlorophyll and suspended matter near the coast of Norway, primarily close to small islands. It is unclear if these estimates are valid or misclassifications since they lie adjacent to pixels marked as invalid in the Sentinel-3 datasets, either due to being clouds or land. Because of the low concentrations estimated by Sentinel-3, it is hard to compare spatial features between the Sentinel-3 and HYPISO-1 maps.

Despite these issues, there are some HYPISO-1 change maps that have spectra similar to chlorophyll. These include:

- Spatial stacking component 6 and 8 (Figure 4.43 and Figure 4.44)
- Spectral differencing and stacking component 10 (Figure 4.50)
- ATGP-generated un-mixing matrix init. match 3 (Figure 4.55)

Each of these results share similar spectral characteristics and have peaks (keeping in mind that some have a reversed sign) around 450 nm and 700 nm, which are associated with the chlorophyll absorption spectrum. Based solely on these spectra, however, it is difficult to determine if algae is detected in the Frohabet site.

The issue with cloud formations dominating many of the components and change maps also appeared in the Frohabet site. A cloud formation at the top of the captures on frequently manifested as an interfering signal in components or its own component in many of the results. A clear example of the cloud formation as a change map appears in the spectral differencing component 1 (Figure 4.47).

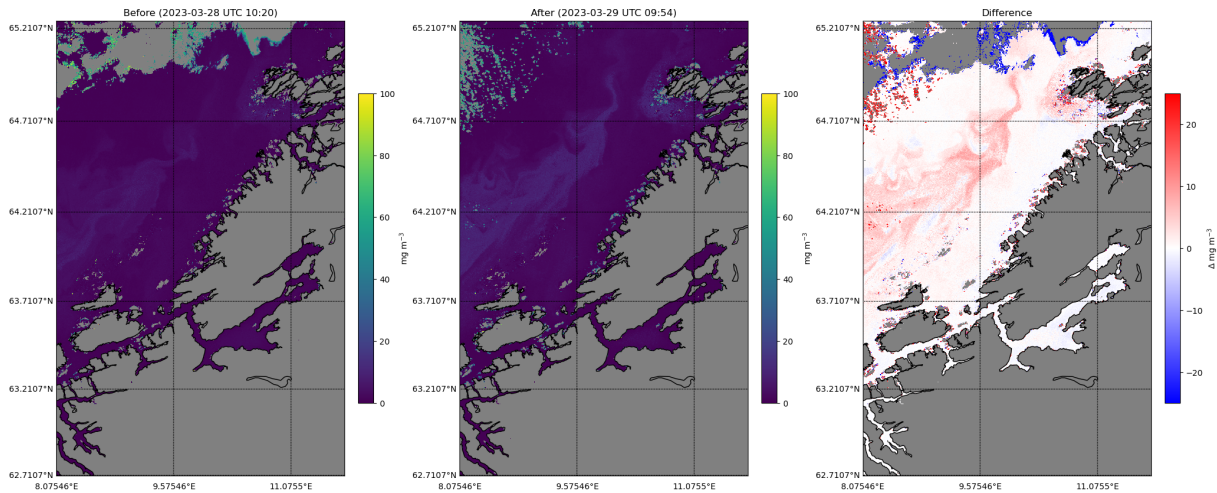


Figure 5.7: Sentinel-3 Chl-a estimates for Frohabet.

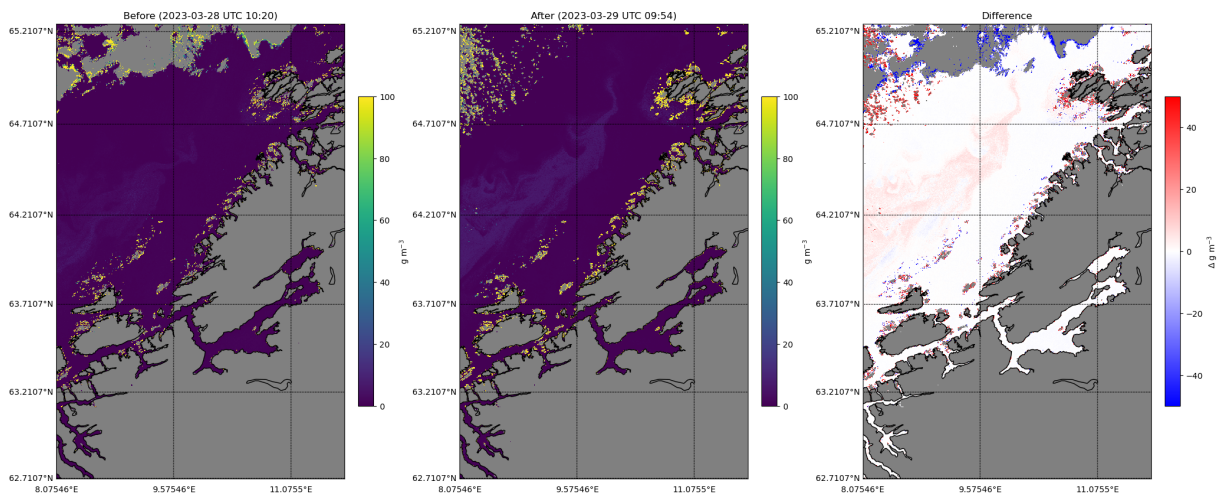


Figure 5.8: Sentinel-3 TSM estimates for Frohabet.

5.2.2 Comparison to NOAA HAB Forecast

The NOAA HAB forecast data serves as a validation dataset for only the Lake Erie target. The NOAA HAB forecast data is derived from Sentinel-3 OLCI observations and provides estimates of algae presence as a relative and dimensionless quantity called the chlorophyll index (CI). The NOAA HAB forecast is generated daily by NOAA for Lake Erie and the western Lake Erie basin regions. The validation

data was already processed into GeoTIFF image plots depicting the chlorophyll index. These images were derived into a change map using QGIS. More background on the NOAA HAB forecast is provided in Section 3.7.

Lake Erie

The NOAA HAB forecasts of Lake Erie are shown in Figure 5.10. The time difference between the data in forecast and the HYPSON-1 captures is shown in Table 5.5. The Lake Erie results are presented in Section 4.3.

Table 5.5: Image acquisition times of HYPSON-1 compared with the times from the NOAA Lake Erie HAB forecasts.

HYPSON-1	NOAA HAB Forecast	Difference
19 July 2022 15:50 UTC	19 July 2022 15:04 UTC	-46 minutes
27 August 2022 16:05 UTC	27 August 2022 15:54 UTC	-11 minutes

In the two NOAA HAB forecasts in Figure 5.10, it can be seen that the Lake Erie algal bloom expanded between 19 July and 27 August 2022. This change was illustrated in Figure 5.9, created by subtracting the chlorophyll index values from Figure 5.10b and Figure 5.10a. The change map in Figure 5.9 only depicts the western basin of Lake Erie.

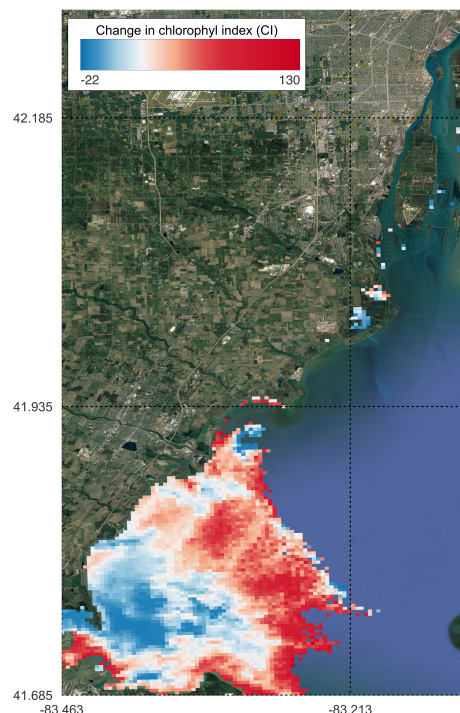


Figure 5.9: Change map computed from the Lake Erie NOAA HAB forecasts dated 19 July 2022 15:04 UTC and 27 August 2022 15:54 UTC. Plotted in QGIS.

The change map in Figure 5.9 shares a similar spatial pattern with the Sentinel-3 OLCI Chl-a and TSM change maps shown in Figure and Figure , respectively. In Figure 5.9, a reduction in the chlorophyll index is observed in the center of the basin while an intensification of the chlorophyll index

is observed near the southern shore as well as further out in the lake to the east. The same general spatial pattern is seen in the Lake Erie results in Section 4.3, including:

- Spatial stacking component 7 (Figure 4.2)
- Spectral stacking component 7 (Figure 4.4)
- Spectral differencing component 4 (Figure 4.5)
- Spectral differencing and stacking component 5 (Figure 4.7)
- Random un-mixing matrix init. match 3 (Figure 4.9)
- Previous un-mixing matrix init. match 3 (Figure 4.11)
- ATGP-generated un-mixing matrix init. match 4 (Figure 4.14)

The fact that the same spatial pattern is observed in the HYPSON-1, Sentinel-3, and NOAA HAB forecast data strengthens the evidence that the anomaly in the Lake Erie scene is caused by algae. Still, determining the exact spectrum of the algae is challenging due to the varying spectra in the different Lake Erie change maps. The discussion related to the spectral interpretation of these HYPSON-1 change maps is covered in the Lake Erie Sentinel-3 validation data comparison in Section 5.2.1.

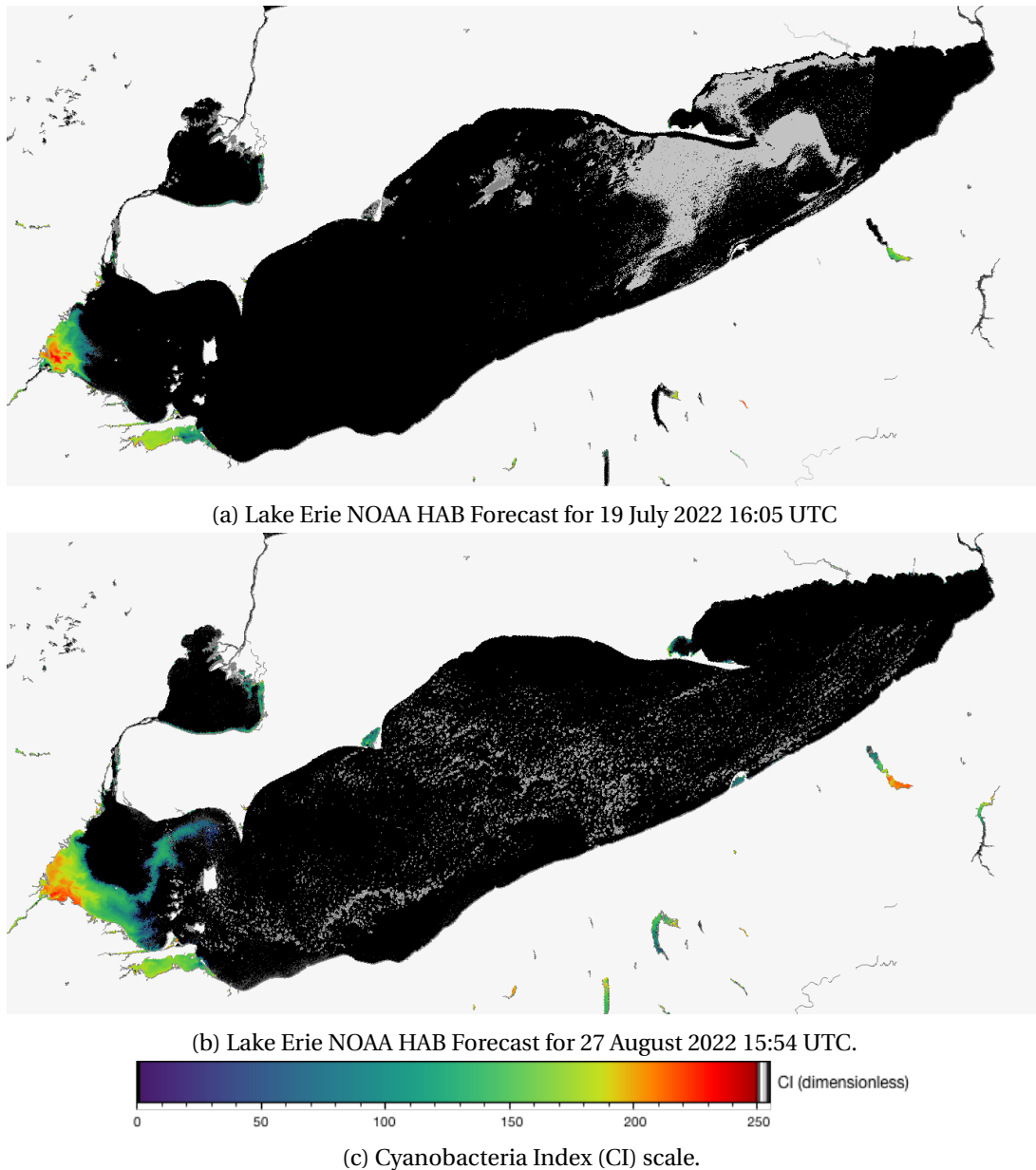


Figure 5.10: Lake Erie NOAA HAB Forecasts

5.2.3 Number of Meaningful Change Maps

Overview

Another evaluation approach entails determining the number of meaningful change maps produced by each technique. This process is somewhat subjective and based on an examination of the spatial distribution and spectra of the independent components and change maps. If one specific technique produces a larger number of meaningful components or change maps, then it may be an indication that the technique is more capable for detecting relevant information and changes in hyperspectral data.

Definition

Meaningful change maps include changing spectral or spatial patterns resembling algae, suspended matter, or other potentially relevant features. In contrast, non-meaningful change maps represent changing features possibly related to the presence of clouds, reflections, noise, or sensor artifacts in the hyperspectral data. Distinguishing between these two classes of features is possible through visual examination of the spatial components or change maps and spectral plots. While counting the number of meaningful change maps for the various observational targets, specific attention was made to disregard change maps related to cloud changes. This was important since no cloud mask was applied to the hyperspectral time series datasets.

The number of meaningful and non-meaningful change maps can be influenced by the initial conditions of the ICA and the number of ICA components chosen for generation (in the case of the paper, 10 were selected). The initial conditions of the ICA, dictated by the initial un-mixing matrix, impact how ICA begins to search for the non-Gaussian independent components in the data. Optimizing them can help ICA more efficiently and accurately pick out the components. This is why PCA and ATGP are used to precondition the initial un-mixing matrix used by ICA. Selecting the appropriate number of ICA components is also an important consideration for ICA. If too few components are chosen, significant contributions to the mixed signal may go undetected by ICA. Conversely, opting for too many components can result in the generation of meaningful independent components and change maps arising from signal noise and artifacts from the hyperspectral sensor. The noise and sensor-related independent components lack spectral information about features in lakes or oceans, making them unhelpful and irrelevant for change detection searching for HABs.

Interpretation

Tables 5.6 to 5.9 show the number of meaningful change maps produced by the different ICA-based change detection methods. Because different scenes or targets may have different sets of spectral features, the number of meaningful change maps should only be compared between the ICA-based change detection techniques for the same target.

Care should be also be taken in comparing the number of meaningful change maps produced by the first four change detection techniques (spatial stacking, spectral stacking, spectral differencing, and spectral differencing and stacking) with the number of meaningful change maps produced by component matching. This is because component matching is able to generate a total number of change maps that is greater than the other techniques through different combinations of the independent components. Non-component matching ICA-based change detection techniques always produce a fixed number of 10 change maps since the quantity is directly tied to the number of independent components from ICA.

Lake Erie

Table 5.6: Number of meaningful change maps for Lake Erie.

CD Technique	Total number of change maps	Number of meaningful change maps
Spatial Stacking	10	5
Spectral Stacking	10	4
Spectral Differencing	10	4
Spectral Differencing and Stacking	10	5
Component Matching (Random W)	19	6
Component Matching (Previous W)	19	5
Component Matching (ATGP W)	20	6

Salish Sea

Table 5.7: Number of meaningful change maps for the Salish Sea.

CD Technique	Total number of change maps	Number of meaningful change maps
Spatial Stacking	10	4
Spectral Stacking	10	4
Spectral Differencing	10	2
Spectral Differencing and Stacking	10	3
Component Matching (Random W)	19	5
Component Matching (Previous W)	19	5
Component Matching (ATGP W)	20	5

Danube River Delta

Table 5.8: Number of meaningful change maps for the Danube River Delta.

CD Technique	Total number of change maps	Number of meaningful change maps
Spatial Stacking	10	3
Spectral Stacking	10	4
Spectral Differencing	10	5
Spectral Differencing and Stacking	10	4
Component Matching (Random W)	20	5
Component Matching (Previous W)	19	4
Component Matching (ATGP W)	20	4

Frohavet

Table 5.9: Number of meaningful change maps for the Frohavet.

CD Technique	Total number of change maps	Number of meaningful change maps
Spatial Stacking	10	2
Spectral Stacking	10	2
Spectral Differencing	10	2
Spectral Differencing and Stacking	10	2
Component Matching (Random W)	19	2
Component Matching (Previous W)	20	2
Component Matching (ATGP W)	20	2

5.2.4 PCC Score Distribution**Overview**

Component matching presents a unique case for evaluations as it involves three distinct ICA initialization approaches. This allows for a comparison not only among the ICA-based change detection techniques but also among the different ICA initialization methods themselves. To do these comparisons, objective comparisons are made by utilizing scores from the Pearson correlation coefficient (PCC) scoring process.

Interpretation

The PCC score serves as an indicator of the quality of independent component matches produced by each ICA initialization approach. An ICA initialization approach that yields matched features with higher PCC scores compared to other initialization approaches implies that the initialization approach aids ICA in identifying the same sets of spectral features in both hyperspectral images. Although higher PCC scores do not necessarily guarantee the relevance of the matched components to the change detection problem, they are nonetheless valuable for evaluating how effectively each initialization approach extracts and matches features.

Table 5.10: Comparison the PCC scores by target.

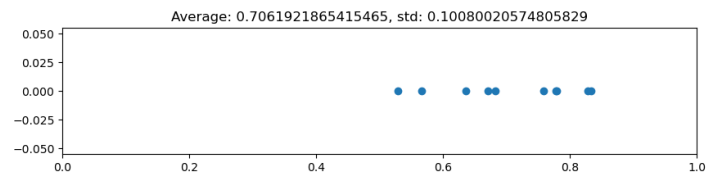
Target	ICA initialization	Score threshold	Number of matches	PCC scores mean	PCC scores standard deviation
Lake Erie	Random W	0.3	19	0.706	0.101
Lake Erie	Previous W	0.3	19	0.716	0.09
Lake Erie	ATGP W	0.3	20	0.745	0.108
Salish Sea	Random W	0.3	19	0.658	0.132
Salish Sea	Previous W	0.3	19	0.662	0.125
Salish Sea	ATGP W	0.3	20	0.663	0.129
Danube River Delta	Random W	0.3	20	0.769	0.115
Danube River Delta	Previous W	0.3	19	0.79	0.117
Danube River Delta	ATGP W	0.3	20	0.792	0.109
Frohavet	Random W	0.3	19	0.778	0.149
Frohavet	Previous W	0.3	20	0.773	0.16
Frohavet	ATGP W	0.3	20	0.747	0.149

Table 5.10 summarized the PCC score statistics from the three different ICA initialization approaches introduced in Section 3.6.10. The values are taken from the actual component matching technique executions used to generate the change maps for the results in Chapter 4. The scores are divided into sections based on the observational targets used in this paper. The component matching techniques were all run with a PCC score threshold of 0.3 to maximize the number of matches. Table 5.12 to Tables 5.14 show the distribution of PCC scores from the results in Chapter 4.

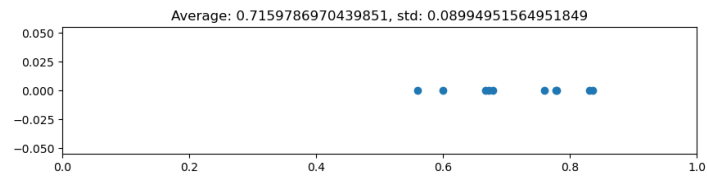
Analysis

All three initialization approaches produced 19 or 20 component matches using ICA configured to produce 10 independent components. Across all four observational targets, the ATGP-generated un-mixing matrix initialization approach consistently produced 20 matches. In terms of the PCC scores, the ATGP-generated un-mixing matrix initialization approach produced the highest PCC score mean for three of the four observational target. Interesting, for the fourth target, Frohavet, the ATGP un-mixing matrix initialization performed the worst, with the random un-mixing matrix initialization approach performing the best.

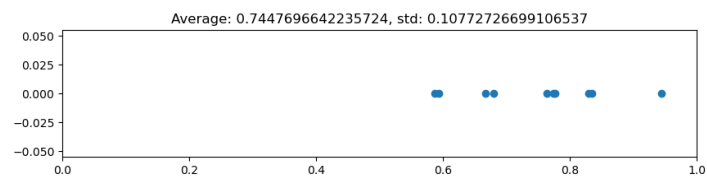
The under-performance of random un-mixing matrix initialization suggests there is a benefit of using information (either in the form of an un-mixing matrix or components) from previous captures to initialize ICA for the next capture in a time series dataset, rather than initializing ICA using randomly selected un-mixing matrix values.



(a) Random un-mixing ICA initialization.

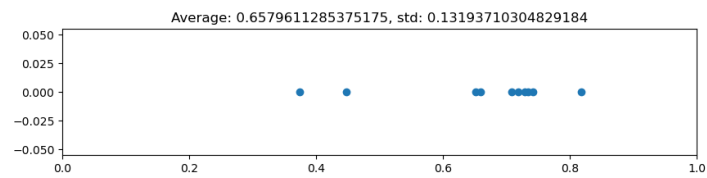


(b) Previous un-mixing ICA initialization.

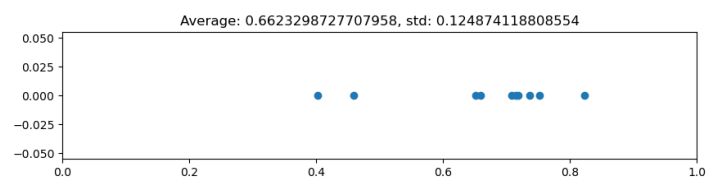


(c) ATGP-generated un-mixing ICA initialization.

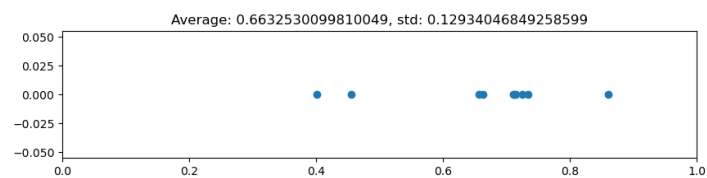
Figure 5.11: Distribution of component matching PCC scores for Lake Erie.



(a) Random un-mixing ICA initialization.

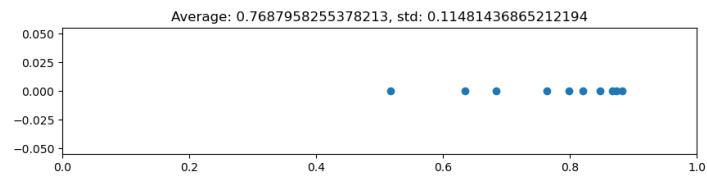


(b) Previous un-mixing ICA initialization.

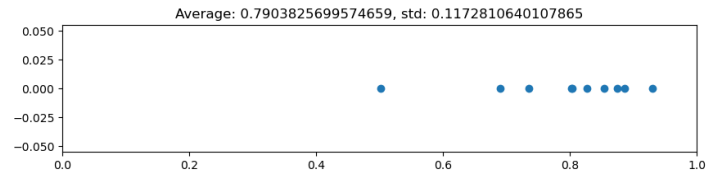


(c) ATGP-generated un-mixing ICA initialization.

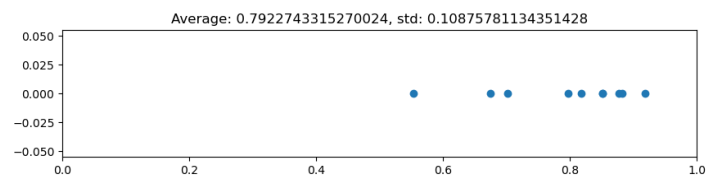
Figure 5.12: Distribution of component matching PCC scores for the Salish Sea.



(a) Random un-mixing ICA initialization.

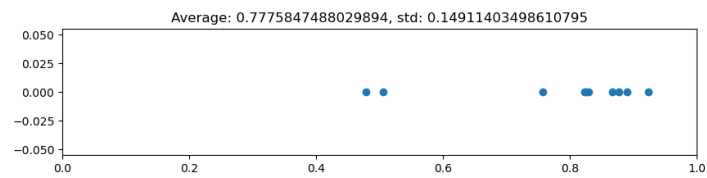


(b) Previous un-mixing ICA initialization.

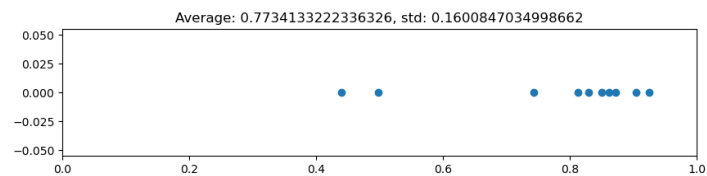


(c) ATGP-generated un-mixing ICA initialization.

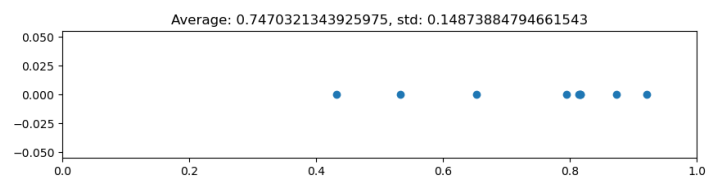
Figure 5.13: Distribution of component matching PCC scores for the Danube River Delta.



(a) Random un-mixing ICA initialization.



(b) Previous un-mixing ICA initialization.



(c) ATGP-generated un-mixing ICA initialization.

Figure 5.14: Distribution of component matching PCC scores for Frohavet.

5.2.5 Interpretability

Overview

The five ICA-based change detection techniques produce distinct sets of change maps, independent component maps, and relative reflectance spectra. Table 5.11 provides an overview of these outputs. An important factor to consider for these ICA-based methods is the interpretability of the generated plots. To effectively be used for detecting HABs or other changing anomalies, the ability to easily interpret and analyze the information presented by the change maps or spectra is critical. This section evaluates the strengths and weaknesses of each technique based on the interpretability of their independent components, change maps, and spectra.

Table 5.11: Comparison of the number of independent component and spectrum sets produced by the ICA-based change detection methods.

CD Technique	Number of IC spatial map sets	Number of IC spectrum sets	Differential spectra?
Spatial Stacking	2	1	No
Spectral Stacking	1	2	No
Spectral Differencing	1	1	Yes
Spectral Differencing and Stacking	1	2	Yes
Component Matching	2	2	No

Spatial Stacking

Spatial stacking produces two sets of spatial maps and one set of spectra. The primary advantage of spatial stacking is its ability to show spatially where an anomaly is detected in the before and after images. By displaying changing anomalies on the before and after maps, it becomes easy to determine the initial and final spatial distributions of an anomaly that changes over time. However, a significant limitation of this method is that it specifically looks for anomalies that exist in both the before and after images. This is why there is only a single relative reflectance spectrum for both the before and after captures in spatial stacking. Consequently, if an anomaly is present in the first capture but disappears before the second capture is taken, it is likely to go undetected using the spatial stacking approach.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Shows spatial distributions of changes • Shows initial and final spatial distributions of anomalies 	<ul style="list-style-type: none"> • Single set of spectra shared by both captures

Spectral Stacking

Spectral stacking operates with one set of spatial maps and two sets of spectra, taking an approach opposite to that of spatial stacking. Unlike spatial stacking, spectral stacking does not display the initial and final spatial distributions of an anomaly in the form of before and after maps. Instead, each component is represented by a single spatial map, which serves as a change map. The trade-off for this difference is that spectral stacking provides less information about the before and after spatial distributions. In exchange, spectral stacking gains information about the before and after states of the relative reflectance spectra through two sets of spectra for each component. This enables spectral stacking to detect anomalies that might appear or disappear in the time between the two images. This trade-off is reflected in Table 5.11, where the spatial stacking technique has two and one sets of spatial maps and spectra, respectively, while the opposite is true for spectral stacking.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Shows spatial distributions of changes • Shows initial and final spectra of anomalies 	<ul style="list-style-type: none"> • Does not show initial and final spatial distributions of anomalies

Spectral Differencing

Spatial differencing generates a single set of spatial maps and spectra. Like spectral stacking, each component is represented by a single spatial map, which functions as a change map. However, spectral differencing primarily showcases the extent to which the spectral values of an anomaly have changed, without directly displaying the before or after spectral values. This is because spectral differencing derives its outputs from a differential datacube. As a result, identifying spectral anomalies is challenging with spectral differencing, especially since there are no static spectral signatures available for analysis. The main advantage of spectral differencing lies in its ability to highlight changes within the data, but it is not well-suited for the direct identification of these changes.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Shows spatial distributions of changes • Shows the change in anomaly spectra 	<ul style="list-style-type: none"> • Does not show initial and final spatial distributions of anomalies • Does not show initial and final spectra of anomalies

Spectral Differencing and Stacking

Spectral differencing and stacking aim to address identification challenges in spectral differencing by incorporating an initial capture datacube into the analysis. As a result, the outputs generated are similar to spectral stacking, composing of one set of spatial maps and two sets of spectra. The main difference here is that the second set of spectra are differential, similar to the ones used in spectral differencing. The first set of spectra originates from the initial capture, enabling the identification of anomalies, while the second set of spectra, being differential spectra, allows for the characterization of detected changes. The spatial maps produced by this approach are change maps, serving to indicate the locations where changes are occurring. However, they do not depict the initial and final spatial distributions of the anomalies themselves and anomalies must be present in the initial capture to be identifiable by their spectra. Overall, spectral differencing and stacking attempts to overcome the limitations of spectral differencing and allowing the identification of changes to be interpreted in its results.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Shows spatial distributions of changes • Shows initial spectra of anomalies • Shows the change in anomaly spectra 	<ul style="list-style-type: none"> • Does not show initial and final spatial distributions of anomalies • Does not show final spectra of anomalies

Component Matching

Component matching generates two sets of spatial maps and two sets of spectra, providing the advantages of displaying both the initial and final spatial distributions and relative reflectance spectra. To overcome the limitation of spectral stacking, which only detects anomalies present in both the before and after images, component matching utilizes a two-step process by running ICA independently for the before and after images. This approach allows for the detection of anomalies in each image separately and produces the maximum number of separate change maps and spectra that detail the captures. However, there is a potential risk of misleading results due to poor component matches. Component matching may attempt to match components from different anomalies, even if they represent different spectral features, leading to false detection and inaccurate interpreta-

tions. Component matching also produces more overall results, which increases the work to find and analysis relevant change maps and spectra.

Advantages	Disadvantages
<ul style="list-style-type: none"> • Shows spatial distributions of changes • Shows initial and final spatial distributions of anomalies • Shows initial and final spectra of anomalies 	<ul style="list-style-type: none"> • Potential for misleading results arising from poor matches • Larger number of change maps and spectra to analyze

5.2.6 Multitemporal Extensibility

Continuous captures of new hyperspectral images by HYPSON-1 contributes to the growth of the HYPSON-1's library of hyperspectral dataset. These hyperspectral images frequently revisit the same locations, enabling them to be used in processed hyperspectral time series datasets. Over time, this expansion ensures the necessary spatial overlap of images for multitemporal change detection. Moreover, the growing dataset enhances the temporal coverage, offering more options for short-term and long-term change detection. The processing pipeline for handling time series data and described in this paper has been developed for the purpose of supporting the creation and expansion of these multitemporal datasets. Bitemporal change detection using ICA and the current hyperspectral datasets has been demonstrated. The successful use of bitemporal ICA-based change detection and growing multitemporal datasets prompts the natural progression towards the multitemporal and trajectory change analysis techniques

Re-purposing the current proposed ICA-based bitemporal change detection techniques to process multitemporal datasets poses challenges. Methods utilizing datacube stacking operations, either in the spatial or spectral dimensions, lead to increased data dimensionality for stacked composite datacubes. While manageable for two datacubes, this approach becomes unsustainable as the number of captures or datacubes increases, demanding more memory and computational resources. To address the issue of increasing dimensionality from data stacking in multitemporal change detection, alternative strategies need to be considered. Subsampling is one such approach that may alleviate some issues with spatial stacking, but it comes at the cost of losing spatial resolution. Another potential strategy is combining redundant spectral bands, similar to the approach taken in the algorithm proposed in S. Liu et al. (2012) [50].

Another option is to avoid stacking methods altogether. The change detection techniques based on differencing and component matching operate on two hyperspectral images at a time, as used for bitemporal change detection. However, these techniques can be successively applied to a series of images, allowing them to be used to perform change detection on multitemporal datasets. This approach enables these techniques to analyze changes over multiple captures, expanding them beyond bitemporal datasets. In multitemporal spectral differencing, the differencing operation could be applied to a series of hyperspectral images. Each subsequent image would be subtracted from the

previous one, generating multiple differential datacubes. These datacubes would then be processed by ICA to produce feature change maps. However, a limitation of multitemporal spectral differencing would be the inability to match the change maps produced from separate differential datacubes and track the evolution of a specific change. This is because the differential datacubes provide information on how the relative reflectance spectrum changes between two images, but they do not offer a static relative reflectance spectrum that can be used for to identify features.

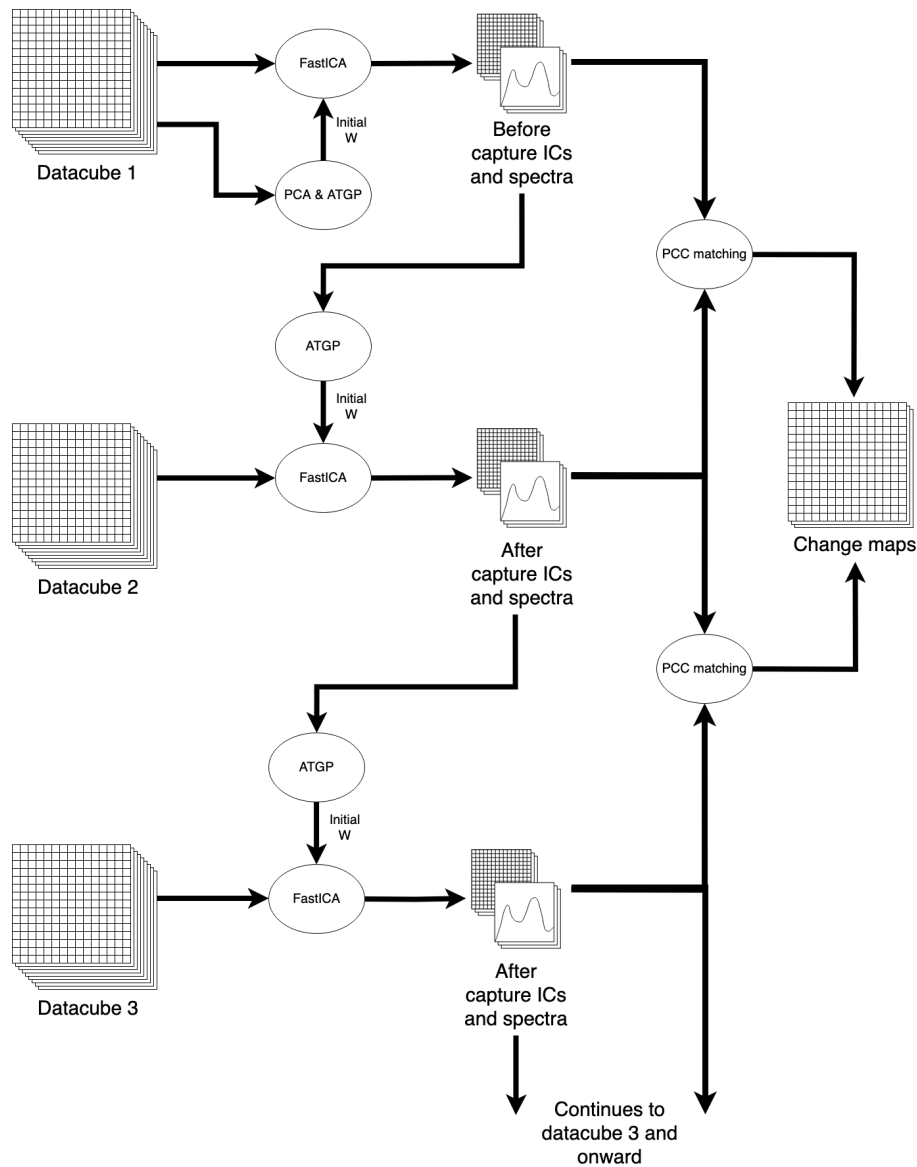


Figure 5.15: Multitemporal ATGP-generated un-mixing matrix ICA initialization approach using previously generated independent components.

Component matching represents another possible non-stacking method to process multitemporal data. To achieve multitemporal component matching, a chained or staged approach could be implemented. The hyperspectral images would be processed two at a time, as is already done for bitemporal change detection, and would be transformed into sets of independent components. The advantage of this approach lies in its immediate transformation of data into independent components, avoiding potential problems related to dimensionality and stacking datacubes. The approach would continue along the same path as the ATGP-generated un-mixing matrix ICA initialization tech-

nique proposed in Section 3.6.10. Each capture in the multitemporal dataset would be initialized using the independent components from the previous capture and using PCA and ATGP to compute the initial un-mixing matrix. In this way, each capture is linked to its immediate predecessor in a continuous chain. The chained approach would effectively extend the original bitemporal ICA initialization approach to a multi-step and multitemporal processing technique. The components generated during each set and pair of images could be matched using PCC and used to produce change maps for features. Since component matching preserves the relative reflectance spectra for the independent component, the spectral signatures could be used to identify features and change maps. Multitemporal component matching is shown as a diagram in Figure 5.15.

These two techniques, spectral differencing and component matching, show that there is a pathway to multitemporal change detection that is based in ICA and the current bitemporal techniques. Alternatively, other dedicated temporal trajectory analysis techniques could also be explored [4]. A notable one involving ICA is spatio-temporal ICA [51], which attempts to extract temporal and spatial components from data.

5.3 Change Detection System

5.3.1 Anomaly Identification

Throughout the comparison between HYPPO-1 results and the validation data, a common theme is the need for better identification of detected anomalies based on their spectral signatures. While ICA offers the ability to identify anomalies based on spectral signatures, the most compelling evidence supporting the detection of algae changes using the ICA-based change detection techniques is based on the similar spatial distributions observed in the HYPOS-1, Sentinel-3, and NOAA HAB forecast data rather than the recovered spectra. A good example of this is the Lake Erie scene. Although certain spectra display characteristics associated with chlorophyll, the variations in spectra make it challenging to precisely pinpoint which spectrum actually represents chlorophyll. This is complicated by the potential presence of other anomalies such as suspended matter or sediments. The inherent ability of ICA to find unexpected anomalies or patterns in the data also contributes to the problem.

Fully understanding and accurately identifying spectra extends far beyond work on change detection. It requires deeper understanding about the biology and spectral characteristics of algae as well as how the spectral characteristics change in different conditions. In this project, most of the identification of algae was made using the assumption that chlorophyll has the two distinct absorption peaks at 450 nm and 700 nm. This approach is inadequate in situations where it is hard to determine if there is a distinct peak present. Additionally, the chlorophyll spectrum can vary based on the environment and conditions, changing the overall spectrum. In the future, more robust identification could be performed using data from in-situ samples and libraries of spectral signatures taken from different materials and substances. Some of this important information could be sourced from other monitoring assets such as autonomous ocean vessels or drones. Furthermore, these activities related to spectral and anomaly identification could be done in collaboration with other groups or partners with specific knowledge about ocean biology as well as familiarity with the needs of the aquaculture industry.

5.3.2 Automation

The primary motivating factor behind the development of ICA-based change detection is to create a novel approach for automatically detecting HABs or aquatic anomalies that could be adapted for operational use. ICA was selected for this purpose because of its unsupervised nature, enabling automatic detection of anomalous signals without the need for labeled data or prior information. Additionally, ICA can analyze hyperspectral images varying in features and conditions. These advantages make ICA a suitable basis for an automated HAB detection system.

An fully autonomous ICA-based change detection system would minimize the need for human involvement in HAB and ocean monitoring. By doing so, resources required for continuous monitoring of coastal regions, such as regular in-situ sampling or sensor network maintenance, could be reduced. The current implementation of ICA-based change detection techniques serves as a proof of concept for this automated detection system and a demonstration of the capabilities of ICA when applied to change detection problems. Operationally, one of the ICA-based change detection techniques would be employed to process both an old and a new capture of the same location whenever a new capture is acquired. The resulting change maps could then be presented to users or used as input for an algorithm to identify the relative reflectance spectra. Any identified changes resembling HABs or targeted anomalies could be flagged for further analysis or necessary actions.

However, several weaknesses still remain that need to be addressed to achieve a fully automated change detection system. For the complete automation, the ability to process bitemporal time series datasets in real time would be necessary. Achieving this would require the issues related to georeferencing, masking, and resampling to be resolved. Determining the significance of independent components and distinguishing them from noise and sensor artifacts is another challenge. While PCA and ATGP are helpful in reducing dimensionality and generating independent components consistently, they do not fully resolve the issue of interpreting independent components. An automated method is needed to differentiate between noise and sensor artifact independent components and the relevant independent components corresponding to ocean spectral signals. This leads to another challenge, which is accurately identifying independent components and change maps in an automated manner. Although ICA-based change detection automatically generates results, their identities may not always be known. This demands the use of either in-situ sampling or a library of spectral reflectance patterns that used to compare and identify spectra from the anomaly and change detection results.

5.4 Improvements to ICA

In this work, FastICA was used to implement and run the ICA algorithm on hyperspectral data. Other than the dimensionality reduction procedure, relatively few changes were made to FastICA to optimize it for anomaly and change detection in hyperspectral images. As such, there still possible routes of further work that could improve the use and functionality of ICA.

5.4.1 Algorithm

Efforts should be made towards a deeper understanding of and characterizing the behavior of ICA. This includes investigating the impact of changing various ICA settings on the components and change maps generated by the ICA algorithm and change detection techniques. Part of this work falls into the

category of ICA algorithm development. Various aspects and parameters of ICA that appear worthwhile to investigate include the number of components, the number of iterations, and as well as modifying the objective function. Investigating these factors may help improve the performance and tailor ICA for the specific task of detecting HABs or other anomalies. There are also ICA implementations other than FastICA that could be trialled and utilized for detection methods.

5.4.2 Capabilities

Other work on ICA should attempt to enhance the capabilities of ICA-based methods. Currently, both the ICA-based anomaly detection technique and the ICA-based change detection techniques lack the capability to estimate algal concentration. Incorporating this capability would offer additional information for the purposes of monitoring for HABs and ocean color analysis. The independent components and change maps produced by ICA show the spatial distribution of anomalies or changes but only provide relative reflectance values and ICA weight values. It would be advantageous to interpret these values as physical quantities that precisely measure the abundance of a specific detected anomaly, much like how the Sentinel-3 OLCI data provide concentrations in units of mass per volume. With these quantities determined, ICA-based change detection techniques could answer the question of exactly how much an anomaly change over a certain period of time rather than simply indicating if an anomalous change is occurring. Further research in this area as well as interpreting ICA results should be pursued.

5.5 Improvements to Time Series Datasets

During the course of this project, the time series dataset generation pipeline was developed from scratch. Although it was successfully used to process and assemble time series datasets of various locations for the ICA-based change detection techniques, it was not necessarily implemented in the most efficient ways. There are still many improvement and changes that could be made to the pipeline, notably to the georeferencing, masking, and resampling steps. The following subsections discuss some of the problems and potential solutions in these areas.

5.5.1 Georeferencing

Currently, HYPSON-1 images are processed with either no or unreliable geospatial spatial information. Proper georeferencing is critical for change detection since the algorithms need to be able to compare and examine the same locations over multiple capture. The dataset pipeline was implemented with a manual georeferencing process; ground control points (GCPs) were selected by hand in QGIS and later loaded into Python. This manual process is a major obstacle for creating time series dataset since it takes time and effort to produced GCPs .points files in QGIS. Additionally, manually selecting GCPs does not work if a HYPSON-1 capture lacks noticeable geographic features, such as the middle of the ocean or a scene with large amounts of cloud cover. As a result, automating the georeferencing step would represent a significant improve to the pipeline. Possible methods of automatic georeferencing include algorithms that factor HYPSON-1's orbit and sensor characteristics and Earth's geometry to determine the field of view and location of a capture. Another possible method is the use of tie points to extend GCPs from one hyperspectral to another hyperspectral image that overlaps it [100].

5.5.2 Cloud Mask

The change maps and independent components are heavily affected by spectral signals from clouds and cloud-covered hyperspectral captures, which dominate more relevant spectral signals. This severely hinders the performance ICA-based change detection, as numerous independent components generated by ICA represent cloud cover spectral features rather than spectral signatures from lakes or oceans on the surface. The purpose of a cloud mask would be to remove or hide cloud cover pixels entirely, similar to what is done in Sentinel-3 OLCI data, and allow spectral signatures from bodies of water to be analyzed by ICA with less interference from undesired spectral signals.

In previous work on ICA anomaly detection [3], a cloud mask for each HYPSON-1 capture was generated using a crude threshold comparison with a unique radiance value selected through trial and error for that capture. A clear weakness of this method is that it requires manual selection of threshold values for each capture, making it unsuitable for an automatic processing pipeline. Furthermore, variations observational conditions and lighting make it impossible to utilize a single threshold value for all hyperspectral captures in a time series dataset and a growing library of images. For these reasons, the development of a robust and automatic cloud mask algorithm should be a priority in order to improve ICA performance. Potential alternatives to the current threshold comparison method include creating a sensor-specific cloud masking algorithm, where only a handful of bands are used. Additionally, support vector machines (SVMs) or naive Bayes classification could also be considered for more effective cloud masking.

5.5.3 Land Mask

In the existing implementation, the land mask used for the time series datasets relies on shoreline information from a shapefile database. However, due to the large file size of the shapefile, it may be beneficial to pursue an alternative approach, such as a dedicated land classification algorithm, to reduce computational demands. It is important that any replacement method has the ability to accurately identify areas in narrow channels or fjords, where fisheries and other aquaculture assets are frequently located. Misclassifications could result in the loss of information in areas where the detection of algae or HABs is most critical. Possible approaches for creating an alternative land mask include methods based on support vector machines (SVMs) or naive Bayes classification.

5.5.4 Resampling

Further improvements could be made to the resampling procedure in the dataset processing pipeline. Resampling is a computationally intensive process and the need to repeat the resampling process 120 times per HYPSON-1 capture highlights a major weakness with the current implementation that uses the Python GeoCube library. Although it can be used with hyperspectral data, GeoCube is primarily developed and intended to handle single channel or RGB data where the resampling process is repeated only a handful of times. However, when applied to hyperspectral data, GeoCube encounters performance issues due to the need to repeat the resampling procedure for tens or hundreds of spectral bands. For example, with HYPSON-1 data, which consists of 120 spectral bands, the resampling process had to be performed 120 times and the processing time for a single capture using GeoCube on a consumer-grade laptop took approximately 10 to 15 minutes. While GeoCube can successfully perform the resampling operation for hyperspectral data, the time required for processing

is considerable, especially when applied to multiple captures that make up a time series dataset. This highlights the need for more efficient resampling tools specifically designed to handle hyperspectral data and the high dimensionality of hyperspectral datasets. Developing efficient resampling tools tailored to hyperspectral data would significantly improve the practicality of processing hyperspectral data, especially for real-time or near real-time applications. One potential solution to the inefficient resampling process is to adapt the code to use the SatPy project's pyresample library. Pyresample is written specifically for satellite imagery and has the built in ability to reuse nearest neighbor calculations, enabling performance improvements where large numbers of spectral bands from the same image are being resampled. [66]

5.5.5 Data Format

The time series data implementation is designed to produce NetCDF files, which offers a reliable file format for handling multidimensional scientific data. However, the set of variables and metadata used internally by NetCDF files is currently not standardized. Standardizing the variable names, bands, and metadata within the NetCDF files would help to maintain consistency and enable better interoperability. The current practice of storing one capture per file proves to be sufficient and simplifies file transfer and also allows for flexible dataset organization, allowing captures to be added or removed from time series datasets. For these purposes, the addition of a manifest file would be beneficial, both for the organization of the captures as well as for users of the datasets.

5.5.6 Other

There is a need for better software tools for manipulating and processing hyperspectral image data. While there are numerous Python packages available, many of them are primarily built for handling multispectral data, leaving a gap for supporting hyperspectral datasets. Developing new or improving existing software and tools to better support hyperspectral data requirements would be a valuable contribution to the wider hyperspectral imaging community. This would simplify data analysis and processing tasks but also make the use of hyperspectral imagery accessible for other research fields such as biology or oceanography.

Chapter 6

Conclusion

6.1 Summary

During this work, the feasibility of ICA-based change detection was demonstrated as a functional method for detecting anomalous changes in bitemporal hyperspectral data. The ICA-based change detection enables multiclass, bitemporal, and feature-based change detection capabilities. Particularly, ICA-based change detection shows potential in identifying and tracking evolving anomalies, such as HABs or the presence of algae in coastal waters.

The success of ICA-based change detection is attributed to several factors. Firstly, the anomaly detection capabilities of ICA enable it to perform unsupervised source signal separation and extract statistically independent features from hyperspectral data, all without prior information about the contents of the data. ICA is based on a deeper level of statistics (non-Gaussianity) than other data transformations such as PCA and can recover spectral information from sources, enabling spectral identification of targets. Additionally, ICA-based anomaly detection is enhanced by a dimensionality reduction method that involves PCA and ATGP, allowing for reliable feature detection within datasets. These properties of ICA and dimensionality reduction are discussed in Chapter 2.

For change detection in bitemporal datasets, the same principles of anomaly detection in single images are extended to more complex time series datasets. ICA is combined with traditional techniques such as data differencing and direct comparison. This combination allows ICA to effectively detect distinct types of changes at a feature level in structured bitemporal data and identify them using the spectra derived from ICA weights, enabling multiclass change detection. Here, the fundamental properties of ICA again allow it to detect changes even without prior information about the nature or location of the changes. The ability to perform change detection automatically without training or needing to know about what changes to expect in a series of hyperspectral images is a valuable aspect of ICA-based change detection methods.

Another contributing factor in the successful demonstration of ICA-based change detection is the newly available datasets of hyperspectral imagery of coastal waters produced by HYPSON-1 since it began its operations. This hyperspectral data has a higher spectral resolution than multispectral data allowing for more subtle spectral features to be detected. Since beginning operations, HYPSON-1 has grown the number of captures that can be used to assemble multitemporal datasets. The unique operational capabilities of HYPSON-1 allow the satellite to revisit and re-observe target at different times, allowing time series datasets covering the same target to be created. Additionally, the foundation for

a software pipeline to create calibrated and georeferenced time series datasets from HYPSON-1 hyperspectral images has been developed. This lays the groundwork for continued work in both bitemporal and multitemporal change detection analysis using HYPSON-1.

6.2 Change Detection Results

ICA-based change detection techniques were applied to bitemporal datasets from diverse locations, including Lake Erie, the Salish Sea, the Danube River Delta, and Frohavet. The results presented in Chapter 4 demonstrate some of the results within the hyperspectral datasets that the change detection techniques produced. The results were compared using validation data from Sentinel-3 and NOAA presented in Chapter 5.

The Lake Erie site yielded promising results, likely detecting a changing algal bloom in the western basin of Lake Erie that was detected by multiple change detection techniques. Using the change detection results, algal bloom was identified using the spatial distribution of the detected change. The HYPSON-1 change maps from the Lake Erie site correlated with same spatial patterns in chlorophyll change maps created using data from both the Sentinel-3 and the NOAA HAB forecast validation sources. Additionally, spectra from the Lake Erie results also provided evidence that the detected changes are algae since they exhibit a distinctive peak around 700 nm.

The Danube River Delta site was another location where a potential change related to algae. In this case, the change was detected in Lake Razim, one of the lake located at the site. Much like the Lake Erie results, the Danube River Delta result from several of the techniques corresponded to spatial patterns in the chlorophyll change maps from Sentinel-3 and also had spectral peaks around 700 nm. In addition to algae, the Danube River Delta site produced results indicating the possibility of detecting changing concentrations of suspended matter or sediments. This observation was seen in Lake Sinoe, another one of the lakes at the site, and corresponded with the TSM change map derived from the Sentinel-3 validation data. However, confirming this detection is challenging due to the lack of information about the specific absorption patterns of suspended matter or sediments.

The other targets, the Salish Sea and Frohavet, produced less certain results. Result from the Salish Sea resembled the spatial distribution of the validation change maps for chlorophyll and TSM but did not have spectral features associated with algae. Frohavet yielded a few interesting spectra, possibly resembling chlorophyll absorption patterns, but they could not be identified due to the lack of chlorophyll and TSM spatial features as indicated by the Sentinel-3 data, except in small localized areas near islands or shores.

Overall, there is evidence that ICA-based change detection techniques were able to detect and map changes in algae. This is primarily from the Lake Erie and Danube River Delta sites where the results corresponded closely with the validation data. There is a possibility of detection the Salish Sea target but the spectral patterns need to be better understood.

6.3 Change Detection Techniques

Across all of the datasets, each of the five ICA-based change detection produced usable change maps indicating where a spectral change occurred. However, not all techniques were equal, particularly when it comes to interpretability. Spatial stacking notably, could only detect features that remained

present in the before and after capture so it would not be suitable for monitoring for phenomena that can suddenly appear. Because of this, it is likely not a good choice for monitoring HABs unless they have already been detected. Other techniques such as spectral stacking or spectral differencing are better suited for detecting unexpected changes.

Spectral differencing is not without its challenges, either. One of its major weaknesses is related to the ability to identify the spectral anomalies shown in its change maps and spectra results since they are derived from a differential datacube. This was a problem when trying to interpret the Salish Sea results in Section 5.2.1. Although a spectral differencing change map resembled Sentinel-3 TSM data, the Salish Sea detection could not be analyzed for its spectrum. For accurate identification of anomalies, spatial and spectral stacking as well as component matching can be more useful and provide more interpretable results. The spectral differencing and stacking technique attempts to reconcile the weaknesses of spectral differencing but also suffers a similar issue as spatial stacking where features need to be present in the first capture to be identifiable.

Component matching is a promising approach to ICA-based change detection and an alternative to the stacking and differencing techniques. It allows spectra from detected changes to be easily examined and it also produces easily interpretable independent component and change maps. Furthermore, component matching presents unique opportunities to adjust the performance of ICA through ATGP and the initialization of the ICA un-mixing matrix. As discussed in Section 5.2.6, component matching also leaves an avenue toward multitemporal change detection that analyzes the change in datasets of more than just two captures.

In summary, each of the ICA-based change detection indicated changes spatially but there is a clear advantage of using non-differencing techniques since they preserve important spectral information that can be used for spectral identification.

6.4 Time Series Datasets

A major prerequisite for the work on the ICA-based change detection was the creation and processing of HYPSON-1 hyperspectral data into usable time series datasets. Building the processing pipeline for this data took a considerable amount of time since many new techniques and tools needed to be learned. Of these, georeferencing and resampling were the most challenging since they had not been done for the HYPSON-1 data. The time series processing pipeline has been developed into a usable tool but there are still improvements to be made, namely the automatic georeferencing of captures as well as more efficient resampling of the hyperspectral data. The development and implementation of a reliable cloud mask is also of key importance. Despite these issues, the pipeline can serve as a starting point for future HYPSON-1 work involving the use of multitemporal data.

6.5 HAB Detection System

In their current state, the ICA-based change detection techniques serve as a proof of concept for an automated detection system capable of monitoring changes in coastal waters. Such a system would have important and useful applications for environmental monitoring and in aquaculture industries. The change maps and spectra derived from hyperspectral time series datasets using ICA-based change detection demonstrate the valuable information that can be extracted and presented using

these techniques. Most importantly, the current implementation successfully utilizes ICA to detect and map changes in algae and other anomalies, as confirmed through the use of validation data.

With these capabilities, ICA-based change detection techniques could in the future serve as a component of an automated or semi-automated HAB detection system, part of a greater observational pyramid. By detecting changing anomalies, these techniques could provide valuable information to identify the occurrence and location of HABs, along with spectral details about type of algae involved. The resulting change maps and spectral data could be directly presented to human analysts for interpretation or further analyzed using other software and used to direct other monitoring assets such as drones or ocean vessels. This would help improve HAB monitoring and responses, enabling timely and accurate detection of harmful algal blooms in coastal waters.

There are still obstacles that need to be overcome to develop such a detection system, however. One major challenge is identifying the spectral features or anomalies involved in detectable changes. Not all spectral features involved in a change can be easily matched to specific physical phenomena or materials such as algae. This can be attributed to various factors, such as unknown spectral signatures, environmental variations, signal noise, or simply divergent spectral patterns captured by ICA. Possible solutions include in-situ sampling to obtain ground truth spectral signatures or cross referencing spectral signatures to library of known spectral signatures to aid in the interpretation of ICA components. In any case, a deeper knowledge of the biology and properties of algae and HABs is essential.

Other efforts aimed at enhancing ICA-based change detection techniques so they could be used operationally include ICA-specific algorithm development in the form of customizing objective functions and other parameters. This could improve accuracy and efficiency of detection. Another area worth investigating is how to derive physical quantities from ICA components such as algal concentration. This would allow ICA-based change detection to determine exactly how severe and how much a HAB is changing over time.

6.6 Final Reflections

In conclusion, ICA-based change detection proves to be a promising technique for monitoring for changes in hyperspectral data at a feature level. It enables the detection, mapping, and analysis of changes based on their spectral signatures. ICA-based change detection has been successfully demonstrated to have detected algae at the Lake Erie site using hyperspectral data from HYPSON-1, which was confirmed using validation data from two separate sources. The work on ICA-based change detection showed five different proof of concept algorithms that could be utilized for future automated HAB detection systems, with component matching showing potential for further development into multitemporal change detection. Furthermore, the work also set the stage for future work involving the use of HYPSON-1 time series data through the development of a time series dataset processing pipeline. Given continued refinement, ICA-based change detection holds the potential become an effective method for hyperspectral change detection, particularly in monitoring HABs and studying changes in the ocean environment using HYPSON-1 data.

Chapter 7

Glossary

ATGP - Automatic Target Generation Process

BIP - Band-Interleaved-by-Pixel

CD - Change Detection

Chl-a - Chlorophyll-a

CI - Cyanobacteria Index

CRS - Coordinate Reference System

CSV - Comma Separated Variable

CVA - Change Vector Analysis

DR - Dimensionality Reduction

ESA - European Space Agency

EVD - Eigenvalue Decomposition

GCP - Ground Control Point

GIS - Geographic Information System

GOCI - Geostationary Ocean Color Imager

GSHHG - Global Self-consistent, Hierarchical, High-resolution Geography Database

HAB - Harmful Algal Bloom

HSI - Hyperspectral Imager

HSTS - HyperSpectral Time Series

HYPSON-1 - Hyper-Spectral Small Satellite for Ocean Observation

IC - Independent Component

ICA - Independent Component Analysis

IMT - Inverse Modelling Technique

IRTM-NN - Inverse Radiative Transfer Model-Neural Network

MBR - Maximum Band Ratio

NaN - Not-a-Number

NASA - National Aeronautics and Space Administration

NCCOS - National Centers for Coastal Ocean Science

NIR - Near Infrared

NOAA - National Oceanic and Atmospheric Administration

NTNU - Norwegian University of Science and Technology

OLCI - Ocean Land Colour Instrument

OSP - Orthogonal Subspace Projection

PCA - Principal Component Analysis

PCC - Pearson Correlation Coefficient

PCC - Post-Classification Comparison

Picard - Preconditioned ICA for Real Data

PRISMA - PRecursores IperSpettrale della Missione Applicativa or Hyperspectral Precursor and Application Mission

RGB - Red, Green, Blue

RX - Reed and Xioli detector

SCV - Spectral Change Vectors

SD - Standard deviation

SDG - Sustainable Development Goals

SNR - Signal-to-Noise Ratio

SSO - Sun Synchronous Orbit

SVM - Support Vector Machine

TSM - Total Suspended Matter

UN - United Nations

References

- [1] Sivert Bakken et al. “HYPSO-1 CubeSat: First Images and In-Orbit Characterization”. In: *Remote Sensing* 15.3 (Jan. 2023). Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, p. 755. ISSN: 2072-4292. DOI: 10.3390/rs15030755. URL: <https://www.mdpi.com/2072-4292/15/3/755> (visited on 04/17/2023).
- [2] Jose M. Bioucas-Dias et al. “Hyperspectral Remote Sensing Data Analysis and Future Challenges”. In: *IEEE Geoscience and Remote Sensing Magazine* 1.2 (June 2013). Conference Name: IEEE Geoscience and Remote Sensing Magazine, pp. 6–36. ISSN: 2168-6831. DOI: 10.1109/MGRS.2013.2244672.
- [3] Cameron Penne. *HYPISO-1 Anomaly Detection with Independent Component Analysis Specialization Project*. 2022.
- [4] Sicong Liu et al. “A Review of Change Detection in Multitemporal Hyperspectral Images: Current Techniques, Applications, and Challenges”. In: *IEEE Geoscience and Remote Sensing Magazine* 7.2 (June 2019). Conference Name: IEEE Geoscience and Remote Sensing Magazine, pp. 140–158. ISSN: 2168-6831. DOI: 10.1109/MGRS.2019.2898520.
- [5] Lucasbosch. *English: Comparison of spectral sampling in RGB, multispectral and hyperspectral imaging*. Feb. 11, 2021. URL: https://commons.wikimedia.org/wiki/File:Spectral_sampling_RGB_multispectral_hyperspectral_imaging.svg (visited on 07/18/2023).
- [6] BRIAN G. WHITEHOUSE and DANIEL HUTT. “OBSERVING COASTAL WATERS WITH SPACEBORNE SENSORS”. In: *Remote sensing of aquatic coastal ecosystem processes*. Ed. by LAURIE L. RICHARDSON and ELLSWORTH F. LeDREW. Remote Sensing and Digital Image Processing. Dordrecht: Springer Netherlands, 2006, pp. 201–215. ISBN: 978-1-4020-3968-3. DOI: 10.1007/1-4020-3968-9_9. URL: https://doi.org/10.1007/1-4020-3968-9_9 (visited on 04/16/2023).
- [7] *EO-1 (Earth Observing-1)*. URL: <https://www.eoportal.org/satellite-missions/eo-1> (visited on 04/16/2023).
- [8] Sanjiwana Arjasakusuma et al. “Change Detection Analysis using Bitemporal PRISMA Hyperspectral Data: Case Study of Magelang and Boyolali Districts, Central Java Province, Indonesia”. In: *Journal of the Indian Society of Remote Sensing* 50.9 (Sept. 1, 2022), pp. 1803–1811. ISSN: 0974-3006. DOI: 10.1007/s12524-022-01566-z. URL: <https://doi.org/10.1007/s12524-022-01566-z> (visited on 01/29/2023).
- [9] *PRISMA (Hyperspectral)*. URL: <https://www.eoportal.org/satellite-missions/prisma-hyperspectral#prisma-mission-phases> (visited on 04/16/2023).

- [10] NASA - NSSDCA - Spacecraft - Details. URL: <https://nssdc.gsfc.nasa.gov/nmc/spacecraft/display.action?id=2018-043A> (visited on 05/24/2023).
- [11] Debra Werner. *Orbital Sidekick acquires first light imagery*. SpaceNews. May 23, 2023. URL: <https://spacenews.com/orbital-sidekick-acquires-first-light-imagery/> (visited on 05/24/2023).
- [12] Sandra Erwin. *NRO signs agreements with commercial providers of hyperspectral imagery*. SpaceNews. Mar. 22, 2023. URL: <https://spacenews.com/nro-signs-agreements-with-commercial-providers-of-hyperspectral-imagery/> (visited on 05/24/2023).
- [13] Mariusz E. Grøtte et al. “Ocean Color Hyperspectral Remote Sensing With High Resolution and Low Latency—The HYPSO-1 CubeSat Mission”. In: *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022). Conference Name: IEEE Transactions on Geoscience and Remote Sensing, pp. 1–19. ISSN: 1558-0644. DOI: 10.1109/TGRS.2021.3080175.
- [14] NTNU SmallSat Lab - NTNU. URL: <https://www.ntnu.edu/smallsat> (visited on 09/06/2022).
- [15] NOAA Great Lakes Environmental Research Laboratory. *Harmful Algal Bloom in Western Basin of Lake Erie: July 9, 2018 (Photo Credit: Aerial Associates Photography, Inc. by Zachary Haslick)*. July 9, 2018. URL: [https://commons.wikimedia.org/wiki/File:Harmful_Algal_Bloom_in_Western_Lake_Erie,_July_9,_2018_\(42544177905\).jpg](https://commons.wikimedia.org/wiki/File:Harmful_Algal_Bloom_in_Western_Lake_Erie,_July_9,_2018_(42544177905).jpg) (visited on 07/19/2023).
- [16] Jorge León-Muñoz et al. “Hydroclimatic conditions trigger record harmful algal bloom in western Patagonia (summer 2016)”. In: *Scientific Reports* 8 (Jan. 22, 2018), p. 1330. ISSN: 2045-2322. DOI: 10.1038/s41598-018-19461-4. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5777999/> (visited on 07/18/2023).
- [17] *June 2023 ENSO update: El Niño is here | NOAA Climate.gov*. URL: <http://www.climate.gov/news-features/blogs/june-2023-enso-update-el-ni%C3%B1o-here> (visited on 07/18/2023).
- [18] Maria Knoph Vignæs. *Slik herjer «dødsalgen» i Nord-Norge*. NRK. Section: nyheter. May 27, 2019. URL: https://www.nrk.no/norge/slik-herjer-_dodsalgen_-i-nord-norge-1.14565942 (visited on 11/22/2022).
- [19] *Lake Erie*. NCCOS Coastal Science Website. URL: <https://coastalscience.noaa.gov/science-areas/habs/hab-forecasts/lake-erie/> (visited on 06/13/2023).
- [20] Hans W. Paerl and Jef Huisman. “Blooms Like It Hot”. In: *Science* 320.5872 (Apr. 4, 2008). Publisher: American Association for the Advancement of Science, pp. 57–58. DOI: 10.1126/science.1155398. URL: <https://www.science.org/doi/10.1126/science.1155398> (visited on 07/18/2023).
- [21] Moderate Resolution Imaging Spectroradiometer. *English: Emiliana huxleyi is about one-tenth the size of a human hair. But in high enough concentrations—a “bloom”—this single-celled phytoplankton becomes visible from space. In May 2020, a vivid bloom of E. huxleyi colored the surface waters of a fjord in southern Norway*. May 20, 2020. URL: https://commons.wikimedia.org/wiki/File:Hardangerfjord_bloom_2020.jpg (visited on 07/19/2023).

- [22] IOCCG. *Phytoplankton Functional Types from Space*. Ed. by S. Sathyendranath. Vol. No. 15. Reports of the International Ocean Colour Coordinating Group. Dartmouth, Canada: IOCCG, 2014. DOI: 10.25607/OBP-106. URL: http://www.ioccg.org/reports/IOCCG_Report_15_2014.pdf.
- [23] John E. O'Reilly et al. "Ocean color chlorophyll algorithms for SeaWiFS". In: *Journal of Geophysical Research: Oceans* 103 (C11 1998), pp. 24937–24953. ISSN: 2156-2202. DOI: 10.1029/98JC02160. URL: <https://onlinelibrary.wiley.com/doi/abs/10.1029/98JC02160> (visited on 04/16/2023).
- [24] JOHN F. SCHALLES. "OPTICAL REMOTE SENSING TECHNIQUES TO ESTIMATE PHYTOPLANKTON CHLOROPHYLL a CONCENTRATIONS IN COASTAL". In: *Remote sensing of aquatic coastal ecosystem processes*. Ed. by LAURIE L. RICHARDSON and ELLSWORTH F. LeDREW. Remote Sensing and Digital Image Processing. Dordrecht: Springer Netherlands, 2006, pp. 27–79. ISBN: 978-1-4020-3968-3. DOI: 10.1007/1-4020-3968-9_3. URL: https://doi.org/10.1007/1-4020-3968-9_3 (visited on 04/16/2023).
- [25] *OC4Me Chlorophyll*. Sentinel Online. URL: <https://copernicus.eu/technical-guides/sentinel-3-olci/level-2/oc4me-chlorophyll> (visited on 07/18/2023).
- [26] Serge Helfrich. *English: Absorption spectrum of chlorophyll a and chlorophyll b*. May 19, 2018. URL: https://commons.wikimedia.org/wiki/File:Chlorophyll_Absorption_Spectrum.svg (visited on 07/19/2023).
- [27] NASA Ocean Biology Processing Group. "GOCI Level 2 Ocean Color Data Version 2014". In: (2016). Publisher: NASA Ocean Biology DAAC. DOI: 10.5067/COMS/GOCI/L2/OC/2014. URL: <http://oceancolor.gsfc.nasa.gov/data/10.5067/COMS/GOCI/L2/OC/2014> (visited on 04/16/2023).
- [28] Changchun Huang et al. "Satellite observation of hourly dynamic characteristics of algae with Geostationary Ocean Color Imager (GOCI) data in Lake Taihu". In: *Remote Sensing of Environment* 159 (Mar. 15, 2015), pp. 278–287. ISSN: 0034-4257. DOI: 10.1016/j.rse.2014.12.016. URL: <https://www.sciencedirect.com/science/article/pii/S0034425714005082> (visited on 04/16/2023).
- [29] *Remote Sensing | Free Full-Text | A Satellite-USV System for Persistent Observation of Mesoscale Oceanographic Phenomena*. URL: <https://www.mdpi.com/2072-4292/13/16/3229> (visited on 07/26/2023).
- [30] Varun Chandola, Arindam Banerjee, and Vipin Kumar. "Anomaly detection: A survey". In: *ACM computing surveys* 41.3 (2009). Place: New York, NY Publisher: ACM, pp. 1–58. ISSN: 0360-0300. DOI: 10.1145/1541880.1541882.
- [31] Przemysław Głomb and Michał Romaszewski. "Anomaly detection in hyperspectral remote sensing images". In: *Hyperspectral Remote Sensing*. Elsevier, 2020, pp. 45–66. ISBN: 978-0-08-102894-0. DOI: 10.1016/B978-0-08-102894-0.00004-8. URL: <https://linkinghub.elsevier.com/retrieve/pii/B9780081028940000048> (visited on 05/23/2023).
- [32] *RXAnomalyDetection*. URL: <https://www.l3harrisgeospatial.com/> (visited on 12/06/2022).
- [33] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern classification*. 2nd ed. New York: Wiley, 2001. 654 pp. ISBN: 978-0-471-05669-0.

- [34] A. Hyvärinen and E. Oja. “Independent component analysis: algorithms and applications”. In: *Neural Networks* 13.4 (June 1, 2000), pp. 411–430. ISSN: 0893-6080. DOI: 10.1016/S0893-6080(00)00026-5. URL: <https://www.sciencedirect.com/science/article/pii/S0893608000000265> (visited on 08/27/2022).
- [35] Pierre Ablin, Jean-François Cardoso, and Alexandre Gramfort. “Faster independent component analysis by preconditioning with Hessian approximations”. In: *IEEE Transactions on Signal Processing* 66.15 (Aug. 1, 2018), pp. 4040–4049. ISSN: 1053-587X, 1941-0476. DOI: 10.1109/TSP.2018.2844203. arXiv: 1706.08171[stat]. URL: <http://arxiv.org/abs/1706.08171> (visited on 10/23/2022).
- [36] *sklearn.decomposition.FastICA*. scikit-learn. URL: <https://scikit-learn/stable/modules/generated/sklearn.decomposition.FastICA.html> (visited on 09/10/2022).
- [37] Jing Wang and Chein-I Chang. “Independent component analysis-based dimensionality reduction with applications in hyperspectral image analysis”. In: *IEEE Transactions on Geoscience and Remote Sensing* 44.6 (June 2006), pp. 1586–1600. ISSN: 0196-2892. DOI: 10.1109/TGRS.2005.863297. URL: <http://ieeexplore.ieee.org/document/1634722/> (visited on 10/01/2022).
- [38] Hsuan Ren and Chein-i Chang. “Automatic spectral target recognition in hyperspectral imagery”. In: *IEEE Transactions on Aerospace and Electronic Systems* 39.4 (Oct. 2003), pp. 1232–1249. ISSN: 0018-9251. DOI: 10.1109/TAES.2003.1261124. URL: <http://ieeexplore.ieee.org/document/1261124/> (visited on 10/17/2022).
- [39] Ian T. Jolliffe and Jorge Cadima. “Principal component analysis: a review and recent developments”. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374.2065 (Apr. 13, 2016), p. 20150202. ISSN: 1364-503X, 1471-2962. DOI: 10.1098/rsta.2015.0202. URL: <https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202> (visited on 12/09/2022).
- [40] Gong Jianya, Ma Guorui, and Qiming Zhou. “A review of multi-temporal remote sensing data change detection algorithms”. In: *Remote Sensing and Spatial Information Sciences, Beijing, China, 3-11 Jul 37* (Jan. 1, 2008).
- [41] Zhenjin Zhou et al. “Change Detection in Coral Reef Environment Using High-Resolution Images: Comparison of Object-Based and Pixel-Based Paradigms”. In: *ISPRS International Journal of Geo-Information* 7.11 (Nov. 2018). Number: 11 Publisher: Multidisciplinary Digital Publishing Institute, p. 441. ISSN: 2220-9964. DOI: 10.3390/ijgi7110441. URL: <https://www.mdpi.com/2220-9964/7/11/441> (visited on 04/16/2023).
- [42] Dengsheng Lu et al. “Change Detection Techniques”. In: *International Journal of Remote Sensing* 25 (Jan. 1, 2004).
- [43] Xiao Benlin et al. “STUDY ON INDEPENDENT COMPONENT ANALYSIS’ APPLICATION IN CLASSIFICATION AND CHANGE DETECTION OF MULTISPECTRAL IMAGES”. In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* XXXVII (2008), p. 871.

- [44] Morton J. Canty and Allan A. Nielsen. “Linear and kernel methods for multivariate change detection”. In: *Computers and Geosciences* 38 (Jan. 1, 2012), pp. 107–114. ISSN: 0098-3004. DOI: 10.1016/j.cageo.2011.05.012. URL: <https://ui.adsabs.harvard.edu/abs/2012CG...38..107C> (visited on 05/29/2023).
- [45] Dennis C. Duro, Steven E. Franklin, and Monique G. Dubé. “A comparison of pixel-based and object-based image analysis with selected machine learning algorithms for the classification of agricultural landscapes using SPOT-5 HRG imagery”. In: *Remote Sensing of Environment* 118 (Mar. 15, 2012), pp. 259–272. ISSN: 0034-4257. DOI: 10.1016/j.rse.2011.11.020. URL: <https://www.sciencedirect.com/science/article/pii/S0034425711004172> (visited on 05/29/2023).
- [46] E TUNG FUNG; LEDREW. “Application of principal components analysis to change detection”. In: *Photogrammetric engineering and remote sensing* (1987). ISSN: 0099-1112.
- [47] Vanessa Ortiz-Rivera, Miguel Velez-Reyes, and Badrinath Roysam. “Change detection in hyperspectral imagery using temporal principal components”. In: *Proceedings of SPIE - The International Society for Optical Engineering*. May 1, 2006. DOI: 10.1117/12.667961.
- [48] B. Jeon and D.A. Landgrebe. “Classification with spatio-temporal interpixel class dependency contexts”. In: *IEEE Transactions on Geoscience and Remote Sensing* 30.4 (July 1992). Conference Name: IEEE Transactions on Geoscience and Remote Sensing, pp. 663–672. ISSN: 1558-0644. DOI: 10.1109/36.158859.
- [49] Francesca Bovolo and Lorenzo Bruzzone. “The Time Variable in Data Fusion: A Change Detection Perspective”. In: *IEEE Geoscience and Remote Sensing Magazine* 3.3 (Sept. 2015). Conference Name: IEEE Geoscience and Remote Sensing Magazine, pp. 8–26. ISSN: 2168-6831. DOI: 10.1109/MGRS.2015.2443494.
- [50] Sicong Liu et al. “Unsupervised hierarchical spectral analysis for change detection in hyperspectral images”. In: *2012 4th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*. 2012 4th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS). ISSN: 2158-6276. June 2012, pp. 1–4. DOI: 10.1109/WHISPERS.2012.6874245.
- [51] Selim Hemissi et al. “A New Spatio-temporal ICA for Multi-temporal Endmembers Extraction and Change Trajectory Analysis”. In: (Jan. 1, 2011).
- [52] *Transforming our world: the 2030 Agenda for Sustainable Development* | Department of Economic and Social Affairs. URL: <https://sdgs.un.org/2030agenda> (visited on 07/15/2023).
- [53] *NumPy*. URL: <https://numpy.org/> (visited on 06/01/2023).
- [54] *SciPy*. URL: <https://scipy.org/> (visited on 06/01/2023).
- [55] *Xarray documentation*. xarray. URL: <https://docs.xarray.dev/en/latest/index.html> (visited on 06/01/2023).
- [56] *pandas - Python Data Analysis Library*. URL: <https://pandas.pydata.org/> (visited on 06/01/2023).
- [57] *Masked arrays — NumPy v1.25 Manual*. URL: <https://numpy.org/doc/stable/reference/maskedarray.html> (visited on 07/14/2023).

- [58] *GeoPandas 0.13.0* — *GeoPandas 0.13.0+0.gaa5abc3.dirty documentation*. URL: <https://geopandas.org/en/stable/> (visited on 06/01/2023).
- [59] *Rasterio: access to geospatial raster data* — *rasterio 1.4dev documentation*. URL: <https://rasterio.readthedocs.io/en/latest/index.html> (visited on 06/01/2023).
- [60] *pyproj 3.5.0 documentation*. URL: <https://pyproj4.github.io/pyproj/stable/> (visited on 06/01/2023).
- [61] *skimage* — *skimage v0.20.0 docs*. URL: <https://scikit-image.org/docs/stable/api/skimage.html> (visited on 06/01/2023).
- [62] *Geocube – Welcome to geocube’s documentation!* — *geocube 0.4.2 documentation*. URL: <https://corteva.github.io/geocube/stable/> (visited on 06/01/2023).
- [63] *Shapely* — *Shapely 2.0.1 documentation*. URL: <https://shapely.readthedocs.io/en/stable/> (visited on 06/01/2023).
- [64] Even Rouault et al. *GDAL*. Version v3.7.0. May 10, 2023. DOI: 10.5281/ZENODO.5884351. URL: <https://zenodo.org/record/5884351> (visited on 06/01/2023).
- [65] *Satpy’s Documentation* — *Satpy 0.42.3.dev0+g2ec18f69.d20230510 documentation*. URL: <https://satpy.readthedocs.io/en/stable/index.html> (visited on 06/01/2023).
- [66] *Pyresample* — *pyresample 1.27.1+0.g12892b8.dirty documentation*. URL: <https://pyresample.readthedocs.io/en/latest/> (visited on 06/25/2023).
- [67] A. Hyvarinen. “Fast and robust fixed-point algorithms for independent component analysis”. In: *IEEE Transactions on Neural Networks* 10.3 (May 1999). Conference Name: IEEE Transactions on Neural Networks, pp. 626–634. ISSN: 1941-0093. DOI: 10.1109/72.761722.
- [68] *Matplotlib* — *Visualization with Python*. URL: <https://matplotlib.org/> (visited on 06/01/2023).
- [69] *matplotlib basemap toolkit* — *Basemap Matplotlib Toolkit 1.2.1 documentation*. URL: https://matplotlib.org/basemap/api/basemap_api.html (visited on 06/01/2023).
- [70] *QGIS – Welcome to the QGIS project!* URL: <https://qgis.org/en/site/> (visited on 06/01/2023).
- [71] *17.3. Georeferencer* — *QGIS Documentation documentation*. URL: https://docs.qgis.org/3.28/en/docs/user_manual/working_with_raster/georeferencer.html?highlight=georeferencer (visited on 06/01/2023).
- [72] *ArcGIS Desktop Help 9.2 - BIL, BIP, and BSQ raster files*. URL: https://webhelp.esri.com/arcgisdesktop/9.2/index.cfm?TopicName=BIL,_BIP,_and_BSQ_raster_files (visited on 04/10/2023).
- [73] Di Tian et al. “Spatiotemporal variability and environmental factors of harmful algal blooms (HABs) over western Lake Erie”. In: *PLOS ONE* 12.6 (June 28, 2017). Publisher: Public Library of Science, e0179622. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0179622. URL: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0179622> (visited on 05/02/2023).

- [74] *NCCOS – Cyanobacteria Algal Bloom from Satellite in Western Lake Erie basin*. NCCOS Coastal Science Website. URL: <https://coastalscience.noaa.gov/science-areas/habs/hab-monitoring-system/cyanobacteria-algal-bloom-from-satellite-in-western-lake-erie-basin/> (visited on 06/04/2023).
- [75] *User Guides - Sentinel-3 OLCI - Sentinel Online*. Sentinel Online. URL: <https://copernicus.eu/user-guides/sentinel-3-olci> (visited on 07/17/2023).
- [76] Irina Catianis et al. “Water Quality, Sediment Characteristics and Benthic Status of the Razim-Sinoie Lagoon System, Romania”. In: *Open Geosciences* 10.1 (Jan. 1, 2018). Publisher: De Gruyter Open Access, pp. 12–33. ISSN: 2391-5447. DOI: 10.1515/geo-2018-0002. URL: <https://www.degruyter.com/document/doi/10.1515/geo-2018-0002/html?lang=en> (visited on 05/02/2023).
- [77] G. Zibordi et al. “In situ autonomous optical radiometry measurements for satellite ocean color validation in the Western Black Sea”. In: *Ocean Science* 11.2 (Mar. 26, 2015). Publisher: Copernicus GmbH, pp. 275–286. ISSN: 1812-0784. DOI: 10.5194/os-11-275-2015. URL: <https://os.copernicus.org/articles/11/275/2015/> (visited on 07/18/2023).
- [78] *BIL, BIP, and BSQ raster files—ArcMap | Documentation*. URL: <https://desktop.arcgis.com/en/arcmap/latest/manage-data/raster-and-images/bil-bip-and-bsq-raster-files.htm> (visited on 02/19/2023).
- [79] *numpy.fromfile — NumPy v1.24 Manual*. URL: <https://numpy.org/doc/stable/reference/generated/numpy.fromfile.html> (visited on 06/12/2023).
- [80] Marie Bøe Henriksen. “Hyperspectral Imager Calibration and Image Correction”. Accepted: 2019-10-31T15:12:18Z. Master thesis. NTNU, 2019. URL: <https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2625737> (visited on 06/12/2023).
- [81] *NTNU-SmallSat-Lab/cal-char-corr: Code for optical calibration, characterisation and corrections*. URL: <https://github.com/NTNU-SmallSat-Lab/cal-char-corr> (visited on 12/08/2022).
- [82] *skimage.transform — skimage 0.21.0 documentation*. URL: https://scikit-image.org/docs/stable/api/skimage.transform.html#skimage.transform.estimate_transform (visited on 06/22/2023).
- [83] *scipy.ndimage.binary_closing — SciPy v1.10.1 Manual*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.binary_closing.html (visited on 06/25/2023).
- [84] *scipy.interpolate.griddata — SciPy v1.10.1 Manual*. URL: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.griddata.html> (visited on 06/25/2023).
- [85] *GSHHG - A Global Self-consistent, Hierarchical, High-resolution Geography Database*. URL: <https://www.soest.hawaii.edu/pwessel/gshhg/> (visited on 03/15/2023).
- [86] *ABI Bands Quick Information Guides*. URL: <https://www.goes-r.gov/mission/ABI-bands-quick-info.html> (visited on 11/22/2022).
- [87] *sklearn.decomposition.PCA*. scikit-learn. URL: <https://scikit-learn/stable/modules/generated/sklearn.decomposition.PCA.html> (visited on 07/12/2023).

- [88] *scipy.stats.pearsonr* — *SciPy v1.11.1 Manual*. URL: <https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.pearsonr.html> (visited on 07/16/2023).
- [89] Kristin Yeager. *LibGuides: SPSS Tutorials: Pearson Correlation*. URL: <https://libguides.library.kent.edu/SPSS/PearsonCorr> (visited on 07/16/2023).
- [90] Igor Ogashawara. “The Use of Sentinel-3 Imagery to Monitor Cyanobacterial Blooms”. In: *Environments* 6.6 (June 2019). Number: 6 Publisher: Multidisciplinary Digital Publishing Institute, p. 60. ISSN: 2076-3298. DOI: 10.3390/environments6060060. URL: <https://www.mdpi.com/2076-3298/6/6/60> (visited on 07/17/2023).
- [91] Timothy Wynne et al. “Relating spectral shape to cyanobacterial bloom in the Laurentian Great Lakes”. In: *International Journal of Remote Sensing - INT J REMOTE SENS* 29 (June 15, 2008), pp. 3665–3672. DOI: 10.1080/01431160802007640.
- [92] Kristy Wallmo. “Harmful Algal Bloom Forecasting Branch Ocean Color Satellite Imagery Processing Guidelines, 2020 Update”. In: (2020). Publisher: National Centers for Coastal Ocean Science (U.S.) DOI: 10.25923/606T-M243. URL: <https://repository.library.noaa.gov/view/noaa/30906> (visited on 10/17/2022).
- [93] *HAB Data Explorer*. HAB Data Explorer. URL: https://products.coastalscience.noaa.gov/habs_explorer/index.php (visited on 07/17/2023).
- [94] *Sentinel-3*. Sentinel Online. URL: <https://copernicus.eu/missions/sentinel-3> (visited on 07/18/2023).
- [95] *Level-2 Water WRR and WFR User Guides - Sentinel-3 OLCI - Level-2 Water - Sentinel Online*. Sentinel Online. URL: <https://copernicus.eu/user-guides/sentinel-3-olci/product-types/level-2-water> (visited on 07/18/2023).
- [96] Nguyen An Binh et al. “Evaluation of Chlorophyll-a estimation using Sentinel 3 based on various algorithms in southern coastal Vietnam”. In: *International Journal of Applied Earth Observation and Geoinformation* 112 (Aug. 1, 2022), p. 102951. ISSN: 1569-8432. DOI: 10.1016/j.jag.2022.102951. URL: <https://www.sciencedirect.com/science/article/pii/S1569843222001480> (visited on 07/18/2023).
- [97] *IMT Neural Net*. Sentinel Online. URL: <https://copernicus.eu/technical-guides/sentinel-3-olci/level-2/imt-neural-net> (visited on 07/18/2023).
- [98] diaswadmin. *Copernicus Sentinel-3 mission - ONDA DIAS*. ONDA-DIAS. URL: <https://www.onda-dias.eu/cms/data/catalogue/sentinel-3/> (visited on 07/18/2023).
- [99] Olivia Lesne and Frédéric Rouffi. “PREPARATION AND OPERATIONS OF THE MISSION PERFORMANCE CENTRE (MPC) FOR THE COPERNICUS SENTINEL-3 MISSION”. In: 2 ().
- [100] Zhenling Ma et al. “Geometric Positioning for Satellite Imagery without Ground Control Points by Exploiting Repeated Observation”. In: *Sensors (Basel, Switzerland)* 17.2 (Jan. 26, 2017), p. 240. ISSN: 1424-8220. DOI: 10.3390/s17020240. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5336085/> (visited on 07/22/2023).

Appendix A

Input File

A.1 Processing Settings Section

Description

The processing settings input file section provides spatial and metadata information required for processing the time series dataset. It also includes input and output paths for data files, calibration files, latitude and longitude spatial bounding box values, as well as file paths to auxiliary information such as the GSHHG cloud mask ESRI shapefile.

Line 1: Time Series Name

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsad IO module.
 - Required: Yes.
 - Allowed Values: 1 or 0.

- TIME_SERIES_NAME
 - Description: Name of time series. Used as part of the filename of the output file.
 - Required: No.
 - Allowed Values: Any string or empty.
 - Default Value: "time_series".

Line 2: Target Name

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsad IO module.
 - Required: Yes.
 - Valid Inputs: 1 or 0.

- TARGET_NAME

- Description: Name of geographic target. Used as part of the filename of the output file.
- Required: No.
- Allowed Values: Any string or empty.
- Default Value: Empty.

Line 3: Output Path

- LINE_FLAG
 - Description: Toggles whether the line is read by the hrad IO module.
 - Required: Yes.
 - Valid Inputs: 1 or 0.
- OUTPUT_PATH
 - Description: Path used as the destination for the processed NetCDF files and the output file.
 - Required: No.
 - Allowed Values: Any Unix-style path.
 - Default Value: "." (working directory.)

Line 4: Spectral Coefficients Path

- LINE_FLAG
 - Description: Toggles whether the line is read by the hrad IO module.
 - Required: Yes.
 - Valid Inputs: 1 or 0.
- SPECTRAL_COEFFS_PATH
 - Description: Path to spectral calibration coefficients file.
 - Required: Yes.
 - Allowed Values: Any Unix-style path.
 - Default Value: Empty.

Line 5: Radiometric Coefficients Path

- LINE_FLAG
 - Description: Toggles whether the line is read by the hrad IO module.
 - Required: Yes.
 - Valid Inputs: 1 or 0.
- RAD_COEFFS_PATH

- Description: Path to radiometric calibration coefficients file.
- Required: Yes.
- Allowed Values: Any Unix-style path.
- Default Value: Empty.

Line 6: Land Mask Path

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsad IO module.
 - Required: Yes.
 - Valid Inputs: 1 or 0.
- LAND_MASK_PATH
 - Description: Path to the GSHHG global land mask ESRI shapefile.
 - Required: No.
 - Allowed Values: Any Unix-style path to a land mask ESRI GIS shapefile (.shp).
 - Default Value: Empty. Land mask will be disabled.

Line 7: Boundary Point 1

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsad IO module.
 - Required: Yes.
 - Allowed Values: 1 or 0.
- LONGITUDE
 - Description: Geographic boundary bottom-left corner longitude in degrees.
 - Required: No.
 - Allowed Values: -180 to 180.
 - Default Value: Empty. Geographic boundary will be calculated from ground control points and geographic extent of the time series captures.
- LATITUDE
 - Description: Geographic boundary bottom-left corner latitude in degrees.
 - Required: No.
 - Allowed Values: -90 to 90.
 - Default Value: Empty. Geographic boundary will be calculated from ground control points and geographic extent of the time series captures.

Line 8: Boundary Point 2

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsd IO module.
 - Required: Yes.
 - Allowed Values: 1 or 0.
- LONGITUDE
 - Description: Geographic boundary top-right corner longitude in degrees.
 - Required: No.
 - Allowed Values: -180 to 180.
 - Default Value: Empty. Geographic boundary will be calculated from ground control points and geographic extent of the time series captures.
- LATITUDE
 - Description: Geographic boundary top-right corner latitude in degrees.
 - Required: No.
 - Allowed Values: -90 to 90.
 - Default Value: Empty. Geographic boundary will be calculated from ground control points and geographic extent of the time series captures.

Line 9: Resampling Resolution

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsd IO module.
 - Required: Yes.
 - Allowed Values: 1 or 0.
- RESAMPLING_RESOLUTION
 - Description: Spatial resolution value used by Geocube to resample the hyperspectral capture data.
 - Required: No.
 - Allowed Values: Any float.
 - Default Value: 0.006.

Line 10: Spectral Band Range

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsd IO module.
 - Required: Yes.

- Allowed Values: 1 or 0.
- START_BAND
 - Description: Spectral band index to start processing from.
 - Required: No.
 - Allowed Values: Any integer.
 - Default Value: Empty. Processing code will use first spectral band available in the data.
- END_BAND
 - Description: Spectral band index to end processing at.
 - Required: No.
 - Allowed Values: Any integer.
 - Default Value: Empty. Processing code will use last spectral band available in the data.

A.2 File Processing Queue Section

Description

Line N>10: Queued File

- LINE_FLAG
 - Description: Toggles whether the line is read by the hsad IO module.
 - Allowed Values: 1 or 0.
- BIP_FILE_PATH
 - Description: Path to the .bip hyperspectral capture file.
 - Required: Yes.
 - Allowed Values: Any Unix-style path to a .bip file.
 - Default Value: Empty.
- GCP_FILE_PATH
 - Description: Path to the .points GCP file.
 - Required: Yes.
 - Allowed Values: Any Unix-style path to a .points GCP file.
 - Default Value: Empty.
- CROSS_TRACK_FLIP_FLAG
 - Description: Toggle to flip capture data across horizontal axis.
 - Required: Yes.
 - Allowed Values: 1 or 0.

- Default Value: Empty or 0 (disabled).
- ALONG_TRACK_FLIP_FLAG
 - Description: Toggle to flip capture data across vertical axis.
 - Required: Yes.
 - Allowed Values: 1 or 0.
 - Default Value: Empty or 0 (disabled).

A.3 Input File Layout

Line	Field 0	Field 1	Field 2	Field 3	Field 4
1	LINE_FLAG	TIME_SERIES_NAME	-	-	-
2	LINE_FLAG	TARGET_NAME	-	-	-
3	LINE_FLAG	OUTPUT_PATH	-	-	-
4	LINE_FLAG	SPECTRAL_COEFFS_PATH	-	-	-
5	LINE_FLAG	RAD_COEFFS_PATH	-	-	-
6	LINE_FLAG	LAND_MASK_PATH	-	-	-
7	LINE_FLAG	LONGITUDE	LATITUDE	-	-
8	LINE_FLAG	LONGITUDE	LATITUDE	-	-
9	LINE_FLAG	RESAMPLING_RESOLUTION	-	-	-
10	LINE_FLAG	START_BAND	END_BAND	-	-
N>10	LINE_FLAG	BIP_FILE_PATH	GCP_FILE_PATH	CROSS_TRACK_FLIP_FLAG	ALONG_TRACK_FLIP_FLAG



 **NTNU**

Norwegian University of
Science and Technology