PONTIFICIA UNIVERSIDAD

CATÓLICA DEL PERÚ

ESCUELA DE POSGRADO

Distributed Hyperspectral Image Analysis based on Cloud Computing Architectures

Tesis para obtener el grado académico de Doctor en Ingeniería que presenta:

VICTOR ANDRES AYMA QUIRITA

Asesor:

CESAR ARMANDO BELTRAN CASTAÑON

Lima, 2022

To God,
for allowing me to wake up each day and look at your greatness.

To my parents, Victor and Rina,
for their example, support and unconditional love.

To Maybee,
my eternal friend, companion, love of my life, your heart is the motivation of all my days.

To Dani and Gabi,
my little scientists, my complete happiness.

Acknowledgment

This has been a long journey, and among this adventure, I am greatly thankful to my advisor, Prof. Cesar Beltran, and to Prof. Gilson Costa, for their sincere friendship and for all their patience, support, understanding, discussions, ideas, and suggestions during my research.

I also thank my colleagues from IA-PUCP and my friends and colleagues from LVC at PUC-Rio, especially Prof. Raul Queiroz, Rodrigo Ferreira, Patrick Happ, and Dário Borges, for their help in the discussions and implementation of previous prototypes of this final product.

I also gratefully acknowledge the financial support of "The Huiracocha Scholarship" and the Pontifical Catholic University of Peru, without which this research would not be possible.

I also thank all my great friends, who have always shown precise support and affection in critical times, you helped me become the person I am today.

I am eternally grateful to Victor and Rina (*in memoriam*), my lovely parents, for all their love, guidance, and support at every stage of my life; and to my dear, intuitive, and awesome siblings, Hugo, Paola, and Isarina, for their patience and love.

Lastly, to my lovely and incredible family; Maybee, for all your love, patience, and support through this prolonged adventure, you are the main piece of my life; to Dani and Gabi, my curious, incredible, delightful, and lovely children, you are the peace, the force, and happiness of my heart.

Abstract

In this thesis, we introduce a novel distributed version of the N-FINDR endmember extraction algorithm, which is able to exploit computer cluster resources in order to efficiently process large volumes of hyperspectral remote sensing data. The implementation of the distributed algorithm was done by extending the InterCloud Data Mining Package capabilities, originally adopted for land cover classification, through the HyperCloud-RS framework, here adapted for performing endmember extraction processes, which can be likewise executed on cloud computing environments, allowing users to elastically access and exploit processing power and storage space within cloud computing architectures, for adequately processing large volumes of hyperspectral data. The framework supports distributed execution, network communication, and fault tolerance, transparently and efficiently to the user. The experimental analysis addresses the performance issues, assessing both accuracy and execution time, over the processing of different synthetic versions of the AVIRIS Cuprite hyperspectral dataset, with 3.1 Gb, 6.2 Gb, and 15.1Gb respectively, thus addressing the issue of dealing with large-scale hyperspectral data. As a further contribution of this work, we describe in detail how to extend the HyperCloud-RS framework by integrating new endmember extraction algorithms within the proposed architecture, thus enabling researchers to implement their own distributed endmember extraction approaches specifically designed for processing large volumes of hyperspectral data.

Keywords: Hyperspectral image processing; Cloud computing, Endmember extraction; Hyperspectral Unmixing; Remote sensing; Large-scale hyperspectral data processing.

El Señor es mi pastor,
nada me faltará.
En lugares de verdes pastos me hace descansar;
junto a aguas de reposo me conduce.
Él restaura mi alma;
me guía por senderos de justicia
por amor de su nombre.

Aunque pase por el valle de sombra de muerte,
no temeré mal alguno, porque tú estás conmigo;
tu vara y tu cayado me infunden aliento.
Tú preparas mesa delante de mí en presencia de mis enemigos;
has ungido mi cabeza con aceite;
mi copa está rebosando.
Ciertamente el bien y la misericordia me seguirán todos los días de mi vida,
y en la casa del Señor moraré por largos días.

Biblia, Salmos, Salmo 23 (Salmo de David)

Contents

List of Figures

List of Tables

Chapter I

Introduction

Since humanity began understanding its environment and the things to which they relate, it initiated the motivation to process the information in order to discover more and new knowledge at every step.

The volume of information that we are currently generating is increasing in exponential ways, according to a report by IBM [1], 2.5 quintillions of bytes of data are now generated every day; remarking that 90% of the data in the world today has been created in the last two years alone; and with new improvements in the devices, sensors, technologies emerging, and heterogeneous data sources, this projection, on the data growth rate, will likely accelerate even more [2, 3, 4, 5].

That is the case when it comes to the remote sensing field, in which, as we entered an era of high-resolution earth observation [6, 7], current estimations show that the remote sensing data gathered by a single satellite data center is increasing in order of Terabytes per day [8, 9, 10], furthermore, over the world, remotely sensed data are now being collected following a Petabyte level growth per day [11]. A large number of earth observation spaceborne and airborne sensors are currently providing massive amounts of remotely sensed data every day, which covers large areas of Earth's surface with unprecedented spectral, spatial and temporal

resolutions [12, 4, 13]. Additionally, the deployment of the latest-generation sensor instruments is providing a nearly continuous stream of high-dimensional and high-resolution remote sensing data [14, 15, 16].

In this sense, during the past few years, improvements on remote sensing systems related to their spatial and spectral resolution and to their revisit frequency, have allowed an increasing remote sensing data availability, providing new information at an extremely fast pace, mainly as a result of recent advances in technologies and sensors for Earth Observation, and to the fact that hundreds of remote sensing satellites are nowadays in orbit acquiring very large amounts of Earth's surface data at every day [16, 17, 18, 19].

For instance, Sentinel-1, from the European Space Agency, generates about 1.5 TB per day [20, 21], the complete Copernicus missions [22], the largest space data provider in the world, currently delivers more than 18.47 TB of daily observations [23], and the NASA's Earth Observation System Data and Information System (EOSDIS) database is experiencing an average archive growth up to 32.8 TB of data per day [24, 25].

In this context, currently handling such large volumes of remote sensing data, in terms of its storage, management, deployment, processing, analysis, and interpretation, impose new and important challenges to be aware of [26, 14, 27, 28, 29, 30], especially regarding computational resources and efficient processing techniques [3, 31, 32, 33]. Furthermore, the manipulation of such large earth observation data can be considered as a big data problem [14, 34, 3], due to the massive amounts of data volumes (TB/day); the resolution (radiometric, spatial, spectral, and temporal), the variety (optical, radar, LIDAR images); the veracity (quality and accuracy of the acquired data), the value (remote sensing application dependent), and the increasing production velocity of information provided by hundreds of multi-resolution remote sensing sensors [14, 35, 36, 37, 4, 38, 39]; problem that is worsened when considering the hyperspectral data scenario [34, 40].

Hyperspectral remote sensing is concerned with the extraction of information from objects or scenes lying on the Earth's surface provided with a high-level spectral resolution, based on the radiance acquired by airborne or spaceborne sensors [41, 42]. Regarding hyperspectral imaging, also defined as imaging spectroscopy [43], the sensor acquires a spectral vector with hundreds of elements for every pixel in the scene, providing the so-called hyperspectral images (HSIs) [13].

Recent advances in hyperspectral imagery, concerning their spatial and spectral resolutions, are continuously enhancing the quality of the information conveyed by them [34]. For instance, in terms of spatial resolution, the Italian PRISMA, the German EnMAP, and the Japanese HySIS orbital systems provide hyperspectral images with up to 250+ spectral bands, at 30 m spatial resolution [44]. Concerning the spectral resolution, advances in hyperspectral sensors currently allow a broad acquisition of spectral bands from the visible up to the short wave infrared, reaching nanoscale spectral resolutions, with narrower bandwidths, as for the Airborne Visible Infrared Imaging Spectrometer (AVIRIS), which acquires information from $400\ \eta$m up to $2400\ \eta$m with its 224 spectral bands [45].

Hyperspectral images are mainly characterized by their high dimensionality and data size, furthermore, hyperspectral remote sensing data represent important sources of information for different applications and scientific research initiatives [46, 47, 48]. Regarding its analysis, in general, hyperspectral image processing is a costly and complex computational process, whose analysis demands efficient computing solutions for scalable and thorough exploitation of the encoded data within this large hyperspectral remote sensing datasets, imposing significant requirements in terms of storage, data processing, and near real-time responses [49, 50, 17, 51, 47, 52, 53].

The most frequently used approach for analyzing hyperspectral remote sensing data is Hyperspectral Unmixing (HU), a process that can be considered a data-intensive computing prob-

lem [47, 54], which provides robust techniques to improve data compression, interpretation, processing, and retrieval, within the context of remote sensing hyperspectral image analysis [55]. More specifically, and according to [53, 56, 13], HU aims at describing the pixels within the hyperspectral image by characterizing their spectral vectors in terms of:

- the spectral properties of the pure components present in the hyperspectral data (also referred to as endmembers); and
- the associated distribution of such endmembers at every pixel in the image (also known as abundance fractions).

Supplementary, HU comprises three main processes [56]: (i) Dimensionality Reduction, usually conducted through Principal Component Analysis (PCA) processing, which is commonly applied given the high spectral correlation in the hyperspectral image; (ii) Endmember Extraction (EE), frequently estimated from the data using geometrical or statistical spectral unmixing approaches; and (iii) Abundance Inversion, which consist in the estimation of the proportions of each endmember at every image pixel. Generally, among those processes, EE is considered the most data-intensive and computing-intensive problem.

Thus, there is an increasing demand for an entire class of techniques, methods, and proper infrastructures for efficient and reliable acquisition, storage, compression, management, access, retrieval, segmentation, interpretation, mining, integration, classification, and visualization of hyperspectral remote sensing data applications, posing new and constant challenges to hyperspectral image analysis [34, 44, 32, 57, 58, 59, 55, 60, 61, 4, 62].

To overcome the aforementioned processing issues, several specialized high-performance computing (HPC) systems have been proposed [63], from multicore-based approaches (exploiting resources from typical desktop computers or workstations) [64, 65], to systems based on graphics processing units (GPUs) [66, 53], field-programmable gate arrays (FPGAs) [67, 68, 69], and computer clusters [47, 70].

However, despite the powerful computing capacities provided by the aforesaid HPC systems, there are still important concerns to be adequately addressed, especially when dealing with large volumes of hyperspectral data and when related to their processing and storage requirements, for which typical HPC systems currently experience some difficulties, even with their enhanced computing capabilities [44].

For instance, although GPUs and FPGAs represent suitable solutions to deal with the near-real-time processing issues, mainly given by their power consumption and onboard processing requirements [67, 71, 72]; when considering large-scale hyperspectral remote sensing datasets, these processing tasks must be performed on distributed computing facilities, such as commodities of clusters [3, 53, 69]. In this sense, homogeneous clusters currently offer access to increased computational power at a low cost in a wide variety of hyperspectral imaging applications [73]; but, as the remote sensing data is continuously increasing, supporting and maintaining such large and single-service-oriented clusters becomes a more demanding and expensive requirement.

Nevertheless, an interesting alternative to the homogeneous clusters is to use highly heterogeneous computing resources [74, 75, 76], where networks of heterogeneous workstations can realize a very high level of aggregate performance computing, furthermore, the pervasive availability of these resources resulted in the current notions of grid and, later, cloud computing infrastructures, which, according to [13], are yet to be fully exploited in processing large-scale hyperspectral imaging problems.

Notwithstanding, as previously described, these GPUs and FPGAs multicores systems struggle with large-scale problems due to their limited memory availability, which is restricted by the amount of data that this type of dedicated hardware may support and process [77]. Additionally, systems based on proprietary physical clusters describe some deficiencies as well, mainly related to traditional data storage mechanisms, high costs of acquisition, installation,

and maintenance, and their low scalability capacity [44].

More recently, as dealing with massive volumes of remote sensing information is becoming a common requirement [44, 32], some researchers started following big data processing trends to overcome the aforementioned issues and began exploiting cloud computing architectures for performing the analysis of large-scale hyperspectral remote sensing data, as presented in [57, 78, 58, 59, 32, 79, 38].

In this regard, cloud computing-based systems offer virtually unlimited capacity for data storage and processing, which can be used to overcome the limitations of other HPC approaches (as the ones mentioned in the previous paragraphs), especially for those related to computing memory availability. On this wise, and within the context of big data processing, cloud computing is a major tendency [27, 80] since it allows the accessing and handling of powerful infrastructures for performing large-scale computing, currently highly demanded because of its dynamic, and on-demand, processing capabilities at reasonable costs [9, 51], thus providing flexible and scalable hardware resources, and lessening user requirements related to purchasing and maintaining complex computing infrastructures [81].

Furthermore, cloud computing not only delivers applications and software as services (SaaS), but also extends its functionalities to the infrastructure and platform as a service; it has led the pay-as-you-go computing and provides the opportunity of accessing infinite computing resources, where users solely pay for the services and resources they use, and probably the most important feature: users do not need to acquire, build, install, and maintain the infrastructure on their own [81].

Considering the remote sensing general scenario, cloud computing can be used as a robust platform for the deployment of big remote sensing data solutions [34, 16, 12, 35, 82, 83, 84], by providing highly scalable storage and high-performance computing capabilities [85, 86, 87], thus becoming a standard for distributed computing due to its advanced capabilities for internet-

scale computing, service-oriented computing, and high-performance computing, offering the potential to tackle massive data processing workloads by means of its distributed parallel architecture [34, 84].

However, according to [48, 57, 51], to the best of our knowledge, and considering the hyperspectral scenario, despite the increasing demand for efficient data processing in the hyperspectral field, there is a limited number of efforts to date, and still not enough operational solutions for exploiting cloud computing infrastructures for large-scale hyperspectral image processing. There are, therefore, still many challenges regarding the integration of cloud computing solutions into hyperspectral remote sensing research, considering that currently cloud computing systems have been shown to perform a high level of aggregate processing in remote sensing applications [48, 44, 47, 4].

In this context, this thesis proposes to design, implement and validate a framework entirely based on cloud computing architectures able to analyze large volumes of hyperspectral data via the endmember extraction process, which is considered the most data-intensive and computing-intensive problem to be solved within the hyperspectral unmixing chain. Furthermore, this framework introduces a novel distributed version of the N-FINDR endmember extraction algorithm, able to perform its analysis on cloud computing resources, in a reliable, scalable, and efficient manner. Additionally, it is expected that the information retrieved with this approach could assist as a valuable resource to cope with real-world application problems, such as data compression, classification processes, anomaly detection, content-based image retrieval (CBIR), among many others, by obtaining relevant insight from such large volumes of hyperspectral data within real remote sensing applications.

It is worth mentioning that programming for cloud environments could represent a complete challenge, and programming models such as MapReduce [88], through its Apache Hadoop open-source implementation [89], and the Pig framework [90], help us to transform this difficult

task into simpler and easier processes to implement, furthermore considering that currently, MapReduce represents one of the main programming models used to process large volumes of data [91], allowing users to focus on data processing rather than abstracting in the details of parallelization, fault-tolerance, data distribution, and load-balancing. In this work, regarding the Pig framework [90], it was adopted for describing the processing structure, as this framework provides the Pig Latin language for expressing data flows, and a compiler for translating Pig Latin scripts into MapReduce jobs [92, 93].

For the validation of the framework and the proposed method, we carried out a series of experiments in which we assessed the accuracy and computation performance of the distributed version of the N-FINDR algorithm for endmember extraction compared to its sequential version; both implementations were executed on different large-scale dataset sizes, that were created as synthetic versions of the well-known AVIRIS Cuprite hyperspectral dataset.

In detail, regarding the accuracy, the endmembers' information, obtained with the sequential and distributed executions of the N-FINDR algorithm, was compared by using the metric proposed in Equation (4.1) [94]. The results demonstrated that regardless of the number of computing nodes used, the same endmember extraction accuracy was obtained: $(\phi_E) = 0.0984$ (being zero the best possible value). This accuracy was also validated in terms of the quality of the image reconstruction process when using the found endmembers, obtaining a mean RMSE value of 2.65×10^{-5} (observing that a low RMSE score corresponds to a high similarity between the original image and the reconstructed one).

Concerning the computation performance, our cloud-based distributed approach achieved high efficiency when processing different dataset sizes, starting with a $2.43\times$ speedup with the smallest dataset (3.1 Gb) when executing the endmember extraction process on a 4 node cluster configuration, and reaching up to $15.81\times$ speedup for the highest dataset (15.1 Gb), when operating with a 32 node cluster configuration.

1.1 Objectives

The main objective of this thesis is to propose and implement a novel distributed approach for the end member extraction process for processing big hyperspectral remote sensing datasets, through a reliable data partitioning scheme and efficient and scalable distributed processing, adequately adapting and exploiting cloud computing architectures for performing hyperspectral image analysis.

In pursuit of the general objective, the specific objectives of this research concentrate on:

- Design and implement a distributed framework based on cloud computing architectures able to process large volumes of hyperspectral data.
- Propose a model for adequately handling the hyperspectral data distribution on this type of architecture, capable to cope with the requirements of spatial-domain partitioning, safeguarding the reliability of the data distribution and the quality of the processes applied on each data partition.
- Modify, integrate and validate the N-FINDR sequential algorithm, which is one of the geometrical-based approaches for endmember extraction process assuming the presence of at least one pure element pixel per endmember, to be able to work on the proposed distributed framework.
- Assess the accuracy and computation performance of the distributed version of the N-FINDR algorithm for endmember extraction on large-scale synthetic versions of the AVIRIS Cuprite hyperspectral dataset.

1.2 Thesis Contributions

This thesis presents a new distributed method for hyperspectral image analysis on big remote sensing datasets based on the hyperspectral unmixing process. Thus, the main contributions of this work are listed below:

- Design and implementation of a novel distributed version of the N-FINDR endmember extraction algorithm [95] built on top of a framework working on a cloud computing environment, which is able to exploit computer cluster resources in order to efficiently distribute and process large volumes of hyperspectral data.
- The implementation of the proposed framework, and its distributed N-FINDR approach, was done by extending the InterCloud Data Mining Package [38] framework, originally adopted for land cover classification. The extended framework, hereinafter referred to as HyperCloud-RS, was adapted here for performing endmember extraction on large volumes of hyperspectral data.
- The proposed HyperCloud-RS framework, which can be executed in different cloud computing environments, allows users to elastically allocate processing power and storage space for effectively handling large amounts of data.
- The HyperCloud-RS framework supports distributed execution, network communication, and fault tolerance, transparently and efficiently to the user, enabling efficient use of available computational resources by scaling them up according to the processing task requirements.
- As a further contribution of this work, it is described in detail how to integrate new endmember extraction algorithms into the HyperCloud-RS framework, mainly targeting those algorithms that belong to the class of pure pixel geometrical-based approaches for

performing linear spectral unmixing, thus enabling researchers to easily implement new distributed approaches for endmember extraction, specifically designed for working on distributed facilities.

- Finally, we implemented an open-source framework able to perform endmember extraction processes on big hyperspectral remote sensing datasets for working on cloud computing infrastructures.

- Lastly, it is worth mentioning that as part of the academic contributions reached during the development of this work, we produced several scientific reports, currently published in Q1 indexed journals and in peer-review conference proceedings as well, as presented in [96, 97, 98, 99]; not mentioning the future research and publications that could be arisen based on this approach, as described in Section 6.1.

1.3 Thesis Organization

The rest of this thesis is organized as follows:

- Chapter II – This chapter presents a comprehensive literature review of the state-of-the-art methods for performing endmember extraction processes on high-performance computing systems, related likewise to the objectives of this thesis.

- Chapter III – Describes the architecture of the proposed HyperCloud-RS framework, furthermore, describes its implementation and presents the distributed approach for the N-FINDR algorithm, as well as describes its fundamentals, and finally presents the guidelines to extend the capabilities of this framework through the inclusion of new endmember extraction algorithms.

- Chapter IV – Describes the dataset used in the experiments, as well as the cloud environment and the particular settings for executing the experimental design. We also describe the metrics used for assessing the accuracy and computational performance for validating the endmember extraction process.

- Chapter V – Presents a detailed discussion of the results achieved with the experiments executed with this framework working on a cloud computing environment.

- Chapter VI – Presents the conclusions of this work along with some directions for future researches.

Chapter II

Related Works

As described in Chapter I, Hyperspectral Unmixing (HU) is the most frequently used approach for analyzing hyperspectral remote sensing data, and the N-FINDR algorithm is among the most frequently used algorithms for the identification of endmembers within the HU processing chain [47]. Thus, this section presents an overview of some related works for endmember extraction for pure pixel geometrical-based approaches, with special attention to N-FINDR parallelized or distributed implementations on high-performance computing infrastructures, when processing medium/large scale hyperspectral datasets.

Since it was first introduced by Winter [95], many different implementations have been proposed for the N-FINDR algorithm [100]. Basically, the algorithm assumes the presence of pure pixels in the original hyperspectral scene, then, through an iterative process that evaluates each pixel in the scene, it tries to maximize the simplex volume that can be formed with the pixel vectors in the data cube. The final vertexes of the largest simplex correspond to the endmembers in the dataset being processed[94]. Such a process represents a very demanding computing task, considering not only the pixel evaluations, but also the amount of information that must be analyzed [47, 77].

2.1 High-Performance Computing Approaches

To overcome the aforementioned endmember extraction processing issues, many alternatives have been proposed, starting from those that try to parallelize the process using multicore architectures, to those that exploit distributed strategies using cluster infrastructures. Currently, there are more sophisticated high-performance computing architectures, allowing the simultaneous use of multiple computing resources and supporting the processing of hyperspectral data on cloud computing platforms, however, to the best of our knowledge, literature still provides few examples of such efforts [32, 47, 73].

Specifically concerning the N-FINDR algorithm, the authors of [64, 65, 101] presented different approaches for performing multi-core processing of the hyperspectral unmixing chain for endmember extraction, providing interesting solutions for parallel approaches of the algorithm. As an evolution of multi-core processing, hardware accelerators became feasible alternatives, for instance, authors in [53, 66, 102, 103, 104, 105] presented perspectives for parallel implementations of the N-FINDR algorithm (and similar endmember extraction approaches) based on Graphics Processing Units (GPU); with a similar objective, authors in [68, 71, 69] introduced some unmixing approaches that use Field-Programmable Gate Array (FPGA).

Previous hardware accelerator approaches achieved approximately near real-time responses in the processing of the hyperspectral data of relative small-medium dataset sizes, being the main limitation/concern of those approaches the amount of data the hardware may efficiently support [34, 77].

According to [47], the most widely used high-performance computing architecture for accelerating hyperspectral-related computations, is cluster computing, where a collection of commodity computers work together, interconnected through a network infrastructure. For instance, the authors of [70, 106, 107, 108, 109] described some cluster-based approaches, where partitioning strategies are required for parallel executing the endmember extraction approaches, thus

the problem is divided into smaller sub-tasks, which are distributed among the cluster nodes. Two types of data partition strategies are used in those approaches, spectral-domain and spatial-domain data partitioning, with the latter being the most frequently investigated so far. However, some major concerns about those solutions are related to the considerable costs involved in the acquisition, implementation, and maintenance of the required computing infrastructure.

2.2 Cloud Computing Approaches

More recently, cloud computing infrastructures have emerged as suitable platforms to overcome the shortcomings of previous high-performance computing methods, mainly considering that cloud computing offers advanced capabilities for service-oriented and high-performance computing [34, 110]. To date, the literature contains few implementations of the N-FINDR algorithm based on cloud computing infrastructures. For instance, in [111], the authors presented a parallelized version of the N-FINDR algorithm implemented on top of the Spark framework, specifically, they actually exploit an advanced feature called broadcast variable abstraction on the Spark engine to implement an efficient data distribution scheme.

Similarly, authors in [44] presented a complete parallel unmixing-based content retrieval system working on cloud computing platforms, and in this work, the endmembers signatures were extracted using the Pixel Purity Index algorithm in parallel, and they achieve interesting speedups when working on large volumes of hyperspectral data.

However, previously, the authors in [112, 113] presented distributed parallel approaches for the identification of endmembers, implementing the Iterative Error Analysis algorithm and the Pixel Purity Index algorithm, respectively, over the Spark framework, using advanced cloud computing technologies to efficiently process massive hyperspectral data; nevertheless, both implementations were tested using relatively small dataset sizes, thus most of the capabilities of such cloud computing implementations could not be fully exploited.

2.3 Related Applications based on HU

Regarding some of the applications, in order to support different applications on hyperspectral imagery, efficient methods for endmember extraction are needed. One of such applications is hyperspectral image classification. For instance, the work [78] describes multi-objective task scheduling for energy-efficient cloud implementation for hyperspectral image classification. In that work, a distributed version of the N-FINDR algorithm is proposed, and the experimental results showed that the multi-objective scheduling approach can substantially reduce the execution time for performing large-scale hyperspectral image classification tasks on the Spark framework.

Another application that requires an efficient implementation of endmember extraction on large hyperspectral image repositories is content-based image retrieval (CBIR). Regarding CBIR applications, and as described in the previous section, the authors of [44, 114] proposed a parallel unmixing-based content retrieval system based on cloud computing infrastructure for assessing a distributed hyperspectral image repository under the guide of unmixed spectral information, extracted using the pixel purity index algorithm, which is a close alternative to the N-FINDR algorithm.

2.4 Chapter Insights

In Table 2.1, we present a summary of the main capabilities and outcomes of the aforementioned parallel/distributed versions of unmixing algorithms, considering the architectures described in this section. The table is not intended to represent a direct comparison of the performances of the different methods and architectures, as the datasets and processing infrastructures vary substantially among implementations; it rather describes some of the characteristics and results delivered by each method, so as to make it possible for the readers to have a general overview

of the capacities and limitations of each solution, either characterized by memory constraints, or non-scalable architectures with high associated costs.

Table 2.1: Summary of the capabilities and outcomes found in the literature for parallel/distributed hyperspectral unmixing methods.

Architecture Type	**Capabilities**			
	Number Nodes/Cores	**Dataset Size**	**Processing Time**	**Operation Costs**
Multicore [64, 65]	From 4 up to 8 cores	50 Mb	Less than 1 s	Installation: Low Maintenance: Low Operation: Free
GPU [66]	From 512 up to 1792 cores	50 Mb	From 4 s to 14 s	Installation: Medium Maintenance: Low Operation: Free
FPGA [68]	-	50 Mb	Less than 1 s	Installation: Medium Maintenance: Low Operation: Free
Cluster [106]	Up to 32 CPUs	140 Mb	50 s	Installation: High Maintenance: High Operation: Free
Cloud [44]	120 cores	Up to 22.4 Gb	4680 s	Installation: Free Maintenance: Free Operation: Low

For instance, the literature related to endmember extraction tasks reports that multicore approaches reached up to the use of eight cores working on 50 Mb small dataset sizes; conversely, GPU implementations largely increased the number of available cores, but both approaches are undermined by the same constraint: memory issues. Moreover, although physical clusters represent an improvement for that matter, they are still constrained by limited memory and low scalability capacity.

Consequently, cloud computing architectures arise as appropriate alternatives to overcome inherent weaknesses of other HPC approaches, as they provide the possibility of using large numbers of computational resources to meet the storage and processing requirements imposed by the big hyperspectral remote sensing data scenario.

It is important to highlight that, and according to [48, 44], there are few works to date describing the use of cloud computing infrastructure for the implementation of remote sensing data processing techniques, in this sense, the search for efficient and scalable solutions for endmember extraction is crucial for creating operational applications, especially those that deal with large hyperspectral datasets.

In this work, as the major contribution to this search, it is introduced a novel distributed version of the N-FINDR algorithm, describing its implementation, which is built on top of a general cloud computing-based framework (HyperCloud-RS framework) for endmember extraction. Moreover, it is further described how new and different endmember extraction algorithms can be implemented within that framework. Finally, it is worth noticing that the proposed distributed implementation was designed to tackle the problem of processing very large volumes of hyperspectral data rather than pursuing major possible speedups through exploiting specific hardware characteristics.

Chapter III

HyperCloud-RS Framework

As described in Chapter I, this thesis proposes the design, implementation, and validation of a framework entirely based on cloud computing architectures able to analyze large volumes of hyperspectral remote sensing data through the analysis of the endmember extraction process. This framework introduces a novel distributed version of the N-FINDR endmember extraction algorithm, able to perform its analysis on cloud computing resources, in a reliable, scalable, and efficient form.

In this sense, this chapter covers the proposed methodology for this implementation and furthermore provides the guidelines for including new endmember extraction algorithms within its architecture, thus extending the framework capabilities.

The first section (3.1) presents the architecture of the HyperCloud-RS framework in detail, which provides a distributed platform for performing endmember extraction processes on cloud environments based on Hadoop and Pig. Section 3.2, describes a particular implementation of the HyperCloud-RS components. Section 3.3 explains the fundamentals of the N-FINDR algorithm, it also presents an interesting analysis regarding the computational complexity of the algorithm and the hardware limitation when processing large volumes of data, then describes the thesis proposal for the distributed version of the N-FINDR algorithm. Finally, Section 3.4

presents the main guidelines to extend the framework capabilities through the inclusion of new endmember extraction algorithms for processing large amounts of hyperspectral remote sensing datasets.

3.1 HyperCloud-RS Architecture

HyperCloud-RS Framework can be regarded as a distributed platform for the interpretation and analysis of large hyperspectral remote sensing datasets. Its architecture was designed for supporting interactions between the algorithms for endmember extraction, operating on large datasets through the MapReduce paradigm, distributing both the data and processing tasks among the machines in a computing cluster connected through a network.

Similar to [38], the architecture of the HyperCloud-RS framework consists of three abstraction layers: project definition; processing; and distribution layer, as depicted in Figure 3.1, which are following described. However, is important to remark that, as compared with the work presented in [38], where the first layer was originally dedicated to pixel-wise classification, here in this approach was modified to enable performing hyperspectral image unmixing.

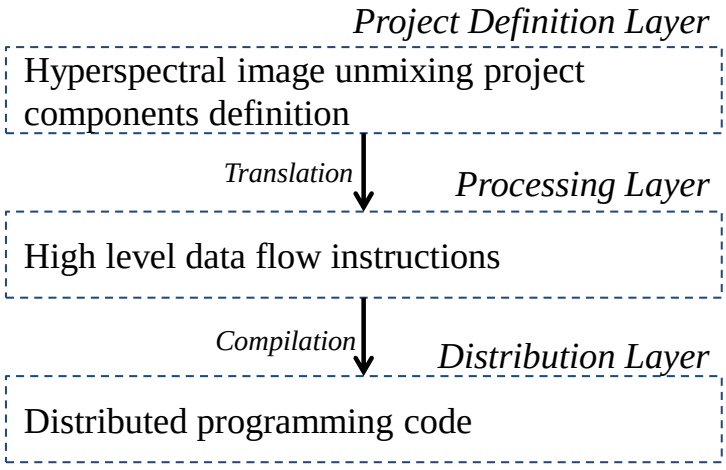

Figure 3.1: HyperCloud-RS architecture.

The project definition layer supports the interaction of end users, that is, specialists that might have no programming skills, but with knowledge about the application (i.e., hyperspectral image analysis). The information provided by end users for this layer comprises the definition of the components of the hyperspectral image unmixing pipeline, which contains all the information required for the execution of the hyperspectral image analysis, namely: the settings for the endmember extraction algorithm definition, the number of processing nodes to be allocated in the cloud computing environment, the repositories for accessing the hyperspectral image dataset and for loading and writing the temporary endmembers' set, and any other cloud-specific computing settings.

The processing and distribution layers remain almost the same as in [38]. The processing layer is meant to be developed by users with regular programming skills to define the project settings and which allows embedding new endmember extraction algorithms into the framework, so that users interacting with the project definition layer would be able to later select among those new algorithm implementations. As an important remark, the translation processes among layers remain unchanged with respect to the implementation in [38], that is, the interaction between the project definition layer and this layer is defined via a translation process, which is responsible for parsing the project definitions settings into processing and data flow instructions, to be coded in a particular high-level programming language, thus masking the complexity of dealing with the distributed programming model.

The distribution layer is responsible for the distributed execution of the hyperspectral image analysis applications, which must be maintained by users habituated with distributed programming models. The interface between the processing layer and this layer is performed via another translation process, which is responsible for decoding the (high-level) instructions, defined in the processing layer, into the required distributed programming instructions to execute the processing applications in a distributed way.

Is important to be aware that these layers contain representations at different levels of abstraction of the processing application. Therefore, to define and execute a particular application, it is required to primary set the lower layers of the framework, referring to the distribution and processing layers, respectively. Then, users could define and execute the respective processing chain through the configuration of the project definition layer.

The main difference between the HyperCloud-RS framework in relation to previous work described in [38] is placed in the distributed processing chain, here adapted for performing distributed endmember extraction processes. In detail, the chain is commenced afterward the parameters of the project definition layer are established, as indicated in Figure 3.2. In sequence, the project settings are translated into data flow instructions, according to the processing layer definitions. Then, the hyperspectral dataset is randomly divided into smaller disjoint subsets, and the endmember extraction algorithm is executed in a distributed way on the processing nodes, as part of the distribution layer processing chain.

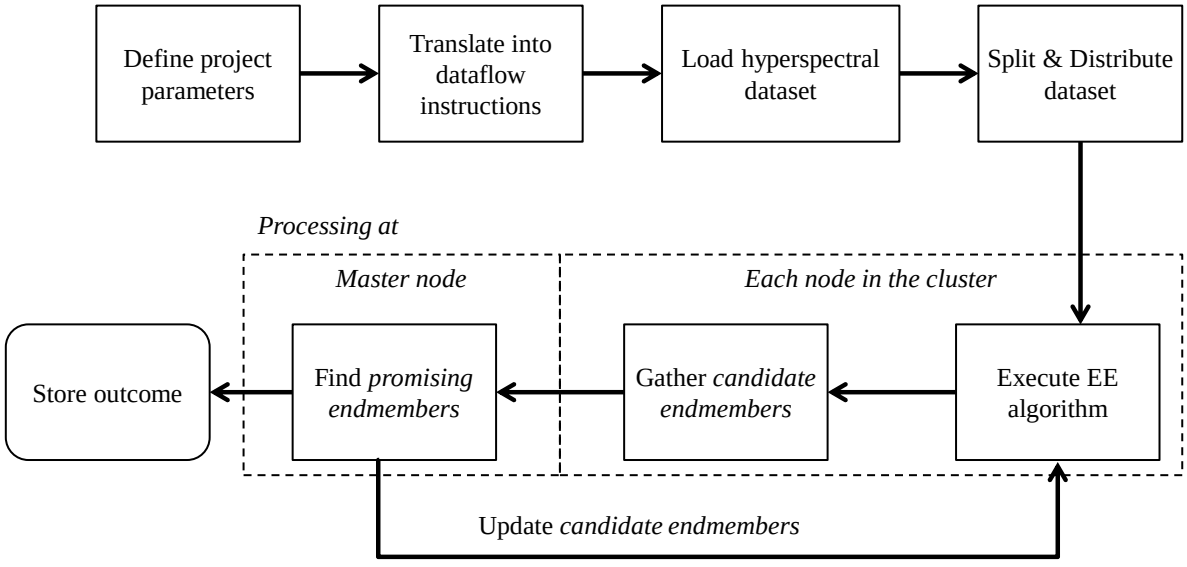

Figure 3.2: General outline of the HyperCloud-RS distributed processing chain.

Following a hierarchical scheme, each hyperspectral subset is handled independently at each

processing node to obtain what is denoted as candidate endmembers (see Section 3.3). Then, the master node gathers such candidate endmembers and processes them using the same endmember extraction algorithm, identifying what we defined as promising endmembers. To ensure the validity of the promising endmembers, they are re-distributed to the cluster, and each processing unit validates that those promising endmembers represent the largest simplex on their local subsets of candidate endmembers. This process is repeated until the maximum volume of the simplex is found, then its vertexes represent the final endmember set for the complete hyperspectral data currently being processed; finally, these outcomes are later stored in a cloud repository.

3.2 HyperCloud-RS Implementation

Following [38], the HyperCloud-RS architecture is implemented through the instantiation of its three abstraction layers and the corresponding translation processes through specific programming frameworks. This section describes a particular implementation of the HyperCloud-RS components.

Apache Hadoop [115], an open-source implementation of the MapReduce [89] distributed programming model, was chosen for instantiating the distribution layer. Currently, Hadoop is one of the most popular frameworks and is widely used for processing very large datasets [91, 115] across nodes in a cluster, which supports processing and data distribution transparently and efficiently to the user [88]. As described in [38], Hadoop has two main components: the distributed file system (HDFS) [116], and the MapReduce programming model [117]. HDFS is designed and optimized for high processing performance and works best with large files (in order of many gigabytes or larger) [116]. Hadoop's MapReduce programming model is based on a simple data processing paradigm composed of three main phases: map, shuffle, and reduce [117].

The Pig framework [90] was adopted for the implementation of the processing layer. Pig was used as an intermediary framework for interfacing the project definition layer with the distribution layer, as it allows the instantiation of custom (or user-defined) functions in a simple way (simpler than using MapReduce directly). This framework provides the Pig Latin language for expressing data flows, and a compiler for translating Pig Latin scripts into MapReduce jobs. Furthermore, Pig Latin language makes programmers' interaction with MapReduce easy, as it is a high-level and extensible programming language through the implementation of its User Defined Functions (UDFs) [93]. As verified in [38], Pig's UDFs provide an extension capacity, allowing the integration of external libraries and scripts (Java, JavaScript, and Python-based) created by third-party developers, providing an easy and efficient way to incorporate new functionalities into the Pig framework.

Finally, the project definition layer was implemented in Java. Through its implementation, the user is able to define all the required settings for the execution of the hyperspectral processing application. Furthermore, Each particular processing algorithm (e.g., endmember extraction) can be structured as a Pig UDF so it can be executed through Pig Latin scripts. Therefore, the proposed architecture supports the addition of new processing algorithms within its structure, so that its capabilities can be easily extended, as described in Section 3.4.

3.3 Endmember Extraction Algorithm

In this research we used the N-FINDR algorithm to validate the performance of the Hyper-Cloud -RS processing chain for the identification of the endmembers in a large hyperspectral remote sensing dataset. In the following subsections we describe the main steps of the N-FINDR algorithm, then we briefly describe its computational complexity constraints, we further introduce the distributed version of N-FINDR, and finally we give the main guidelines for integrating new endmember extraction algorithms with the HyperCloud-RS framework.

3.3.1 N-FINDR Algorithm

The N-FINDR algorithm belongs to the class of pure pixel geometrical-based approaches for performing linear spectral unmixing, assuming the presence of at least one pure pixel per endmember in the input data.

As described in [56], geometrical-based approaches exploit the fact that linearly mixed vectors belong to a simplex set. Furthermore, pure pixel-based algorithms assume that there is at least one pure spectral vector on each vertex of the data simplex. However, this assumption may not hold in many datasets, in which case, those algorithms try to find the set of the possible purest pixels in the data.

The N-FINDR algorithm [95] finds the set of pure pixels defining the largest possible simplex volume by inflating a simplex inside the data in order to identify the endmembers. The endmembers are supposed to be placed in the vertexes of the largest simplex, on the assumption that, among the spectral dimensions, the volume defined by the simplex formed by the purest pixels is larger than any other volume defined by some other combination of (non-pure) pixels [56].

As described in [118], given an initial number of p-endmembers, with the spectral dimensionality of the hyperspectral dataset being transformed to $p-1$ dimensions , the N-FINDR algorithm starts with a random set of p initial endmembers $\left\{\boldsymbol{e}_1^{(0)}, \boldsymbol{e}_2^{(0)}, \ldots, \boldsymbol{e}_p^{(0)}\right\}$, where $\boldsymbol{e}_i$ is a column vector representing the i_{th} endmember spectral values. Then, an iterative procedure is employed to find the final endmembers. As shown in Equation (3.1), at each iteration $k \geq 0$, the volume of the simplex $V\left(\boldsymbol{e}_1^{(k)}, \boldsymbol{e}_2^{(k)}, \ldots, \boldsymbol{e}_p^{(k)}\right)$, is computed as:

$$V\left(\boldsymbol{e}_1^{(k)}, \boldsymbol{e}_2^{(k)}, \ldots, \boldsymbol{e}_p^{(k)}\right) = \frac{\left|\det\begin{bmatrix} 1 & 1 & \ldots & 1 \\ \boldsymbol{e}_1^{(k)} & \boldsymbol{e}_2^{(k)} & \ldots & \boldsymbol{e}_p^{(k)} \end{bmatrix}\right|}{(p-1)!} \tag{3.1}$$

Next, given a sample pixel vector $\boldsymbol{r}$ from the dataset, it is required to calculate the volumes of p simplexes: $V\left(\boldsymbol{r}, \boldsymbol{e}_2^{(k)}, \ldots, \boldsymbol{e}_p^{(k)}\right)$, $V\left(\boldsymbol{e}_1^{(k)}, \boldsymbol{r}, \ldots, \boldsymbol{e}_p^{(k)}\right)$, $V\left(\boldsymbol{e}_1^{(k)}, \boldsymbol{e}_2^{(k)}, \ldots, \boldsymbol{r}\right)$. If none of these p recalculated volumes is greater than $V\left(\boldsymbol{e}_1^{(k)}, \boldsymbol{e}_2^{(k)}, \ldots, \boldsymbol{e}_p^{(k)}\right)$, then the endmember in $\boldsymbol{e}_1^{(k)}, \boldsymbol{e}_2^{(k)}, \ldots, \boldsymbol{e}_p^{(k)}$ remain unchanged; otherwise, the endmember which is absent in the largest volume from the p simplexes is substituted by the sample vector $\boldsymbol{r}$, producing a new set of endmembers. This process is repeated up to all pixel vectors from the hyperspectral dataset are evaluated. The outcome of this process is the mixing matrix $\mathbf{M}$ containing the spectral signatures of the $[\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_p]$ endmembers present in the hyperspectral dataset.

From a geometrical point of view, Figure 3.3 [56] presents a representation of a 2-simplex set C for a hypothetical mixing matrix $\mathbf{M}$ containing $p = 3$ endmembers (considering C as the convex hull of the columns of $\mathbf{M}$). It is worth noticing that the green points represent spectral vectors of the dimensionality reduced hyperspectral dataset, and the red points represent the endmember set in the data. Such a geometrical-based approach is the basis of many other unmixing algorithms.

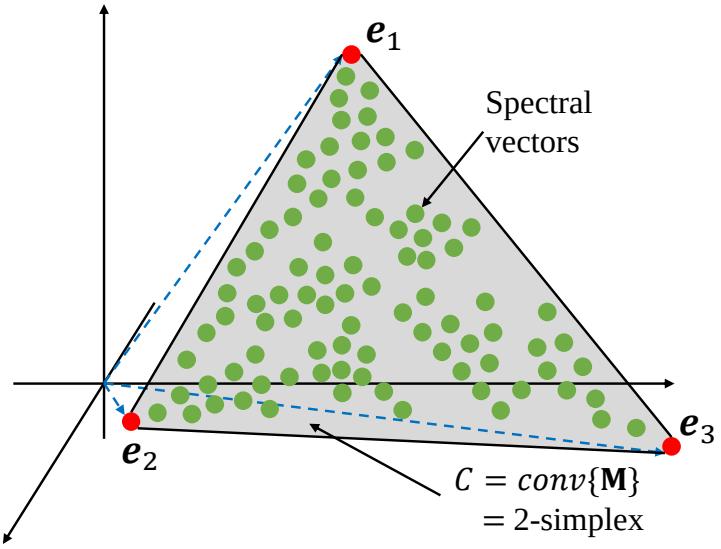

Figure 3.3: Geometrical illustration for the simplex set C for $p = 3$ endmembers. Red circles (vertexes of the simplex) correspond to the endmembers. Green circles represent the spectral vectors.

3.3.2 N-FINDR Computational Complexity

As previously stated, the N-FINDR algorithm is a geometrical-based approach for performing linear spectral unmixing. Let us now consider a larger dataset than that presented in Figure 3.3, where the number of spectral vectors is exponentially increased; applying the original (sequential) N-FINDR algorithm for finding the endmembers in this new and larger dataset will be an extremely time-consuming task, and sometimes could even represent an impossible process to perform, depending on the dataset size and the limitations of the hardware capabilities.

To introduce an important overview regarding the computational complexity of the N-FINDR algorithm, Equation (3.2) provides such complexity value (C), whose growth rate is mainly driven by the number of endmembers to compute (p), and the number of pixels to assess (N). In this regard, Figure 3.4, presents the floating point operations required by the algorithm to provide 1, 2, 3, and up to 10 endmembers when assessing different numbers of pixels.

$$C = N * p * \left(\frac{2}{3}p^3 - \frac{1}{2}p^2 + \frac{5}{6}p - 1\right) \tag{3.2}$$

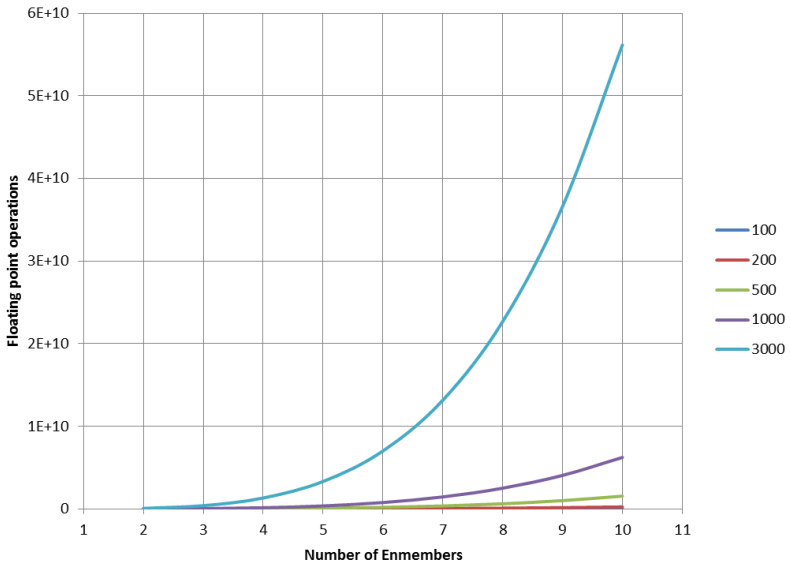

Figure 3.4: N-FINDR algorithm computational complexity

In Figure 3.4, it can be seen that the complexity of the N-FINDR algorithm exponentially increases as the number of endmembers and pixels to be assessed increases as well, which (as described before), depending on the specific computing hardware capabilities, could represent a computational complex, or even impossible, process to perform.

Additionally, regarding the dataset size concern, Figure 3.5 shows an interesting assessment of the processing times required to perform the N-FINDR endmember extraction process when working on different datasets sizes (described in terms of Mb in the figure) for a given number of endmembers ($p = 9$). Besides, it is important to remark that these outcomes were provided by a standalone machine equipped with an Intel(R) Core(TM) i7-3612QM (2.10GHz) processor with 4 CPUs, and with 16 Gb of RAM, executing a sequential implementation of the N-FINDR algorithm.

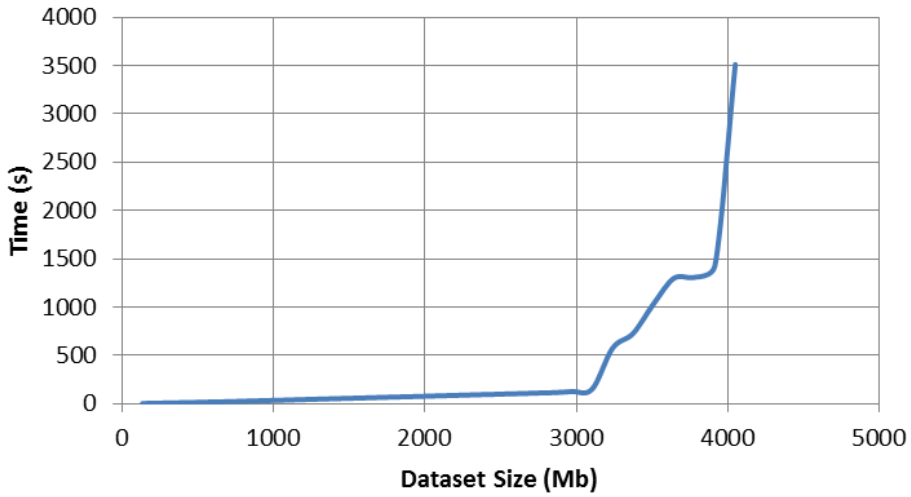

Figure 3.5: Computational limits for processing the N-FINDR algorithm

As can be observed in Figure 3.5, the processing time of the N-FINDR algorithm is quite low, up to certain dataset size, of around 3000 Mb; however, as the dataset size increases, the processing time experience almost exponential growth, once again, up to a certain limit of around the 4000 Mb, after which the computing machine completely collapse.

Therefore, to tackle these obstacles, we propose a novel distributed version of the N-FINDR algorithm, further described in the following section.

3.3.3 Distributed N-FINDR Algorithm

The proposed method is based on a master-slave computing approach, tailored to be executed in a computer cluster. The design of the method takes into consideration the nature of the geometrical-based endmember extraction techniques. The main idea is to perform a preprocessing step at the master computing node consisting of performing a random disjoint data partitioning, which will enable the processing of each data subset independently in the slave nodes, each executing our distributed version of the N-FINDR algorithm. Then, after the endmembers associated with each subset are found, only those data points, defined as candidate endmembers, are submitted back to the master node, which will execute the N-FINDR algorithm again, but solely over those candidate endmembers.

To better illustrate the proposed method, let us assume that we have a large hyperspectral dataset, and for exemplification purposes consider the data is transformed into a lower spectral dimensionality of two dimensions, with $p = 3$ endmembers, as presented in Figure 3.6a. As described before, the first step is to perform a random partition of the complete dataset, as illustrated in Figure 3.6b, where each color describes different data partitions. It is important to notice that the number of partitions/subsets created should be equal to or larger than the number of processing nodes in the cluster to ensure: i) substantial performance improvements, and ii) no idle processing nodes. Afterward, the subsets are distributed among the slave nodes and each one executes the N-FINDR algorithm, finding the vertexes of the simplexes within its own data subset, which are defined as the candidate endmembers, providing one set of candidate endmembers per partition, as presented in Figure 3.6c, where the analysis of each partition in the figure provides 3 vertexes defined as their candidate endmembers.

Later, the candidate endmembers are gathered at the master node, to subsequently be re-processed with the N-FINDR algorithm to obtain what we defined as the promising endmembers, represented in red circles in Figure 3.6d, which will be submitted back to the slave nodes for performing a validation process. In this validation process, each slave node verifies that its candidate endmembers subset is contained within the simplex defined by the promising endmembers, finally providing the complete set of endmembers of the full hyperspectral dataset.

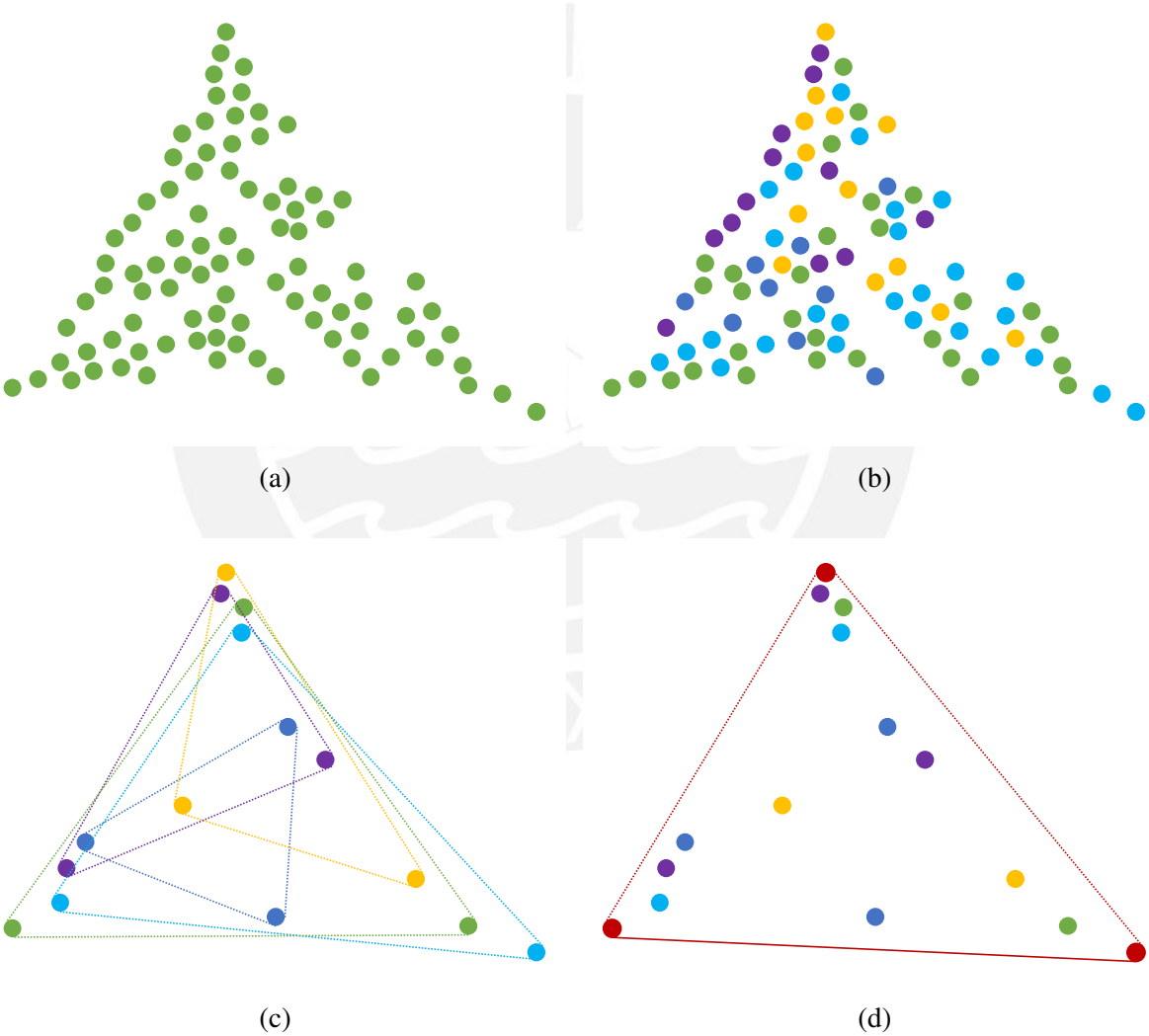

Figure 3.6: Distributed N-FINDR Algorithm processing scheme: (**a**) Geometrical illustration of the dataset for $p = 3$ endmembers, (**b**) Random partitioning of the dataset, (**c**) Simplexes found at each processing node, (**d**) Promising endmembers processed at the master node.

In case the validation step fails, the complete process should be repeated until the promising endmembers met the validation process, or a maximum number of predefined iterations is completed, thus providing the set of final endmember. An important remark is that at each new iteration the promising endmembers are taken as the set of initial endmembers for each new execution of the N-FINDR algorithm within each slave node.

Finally, as observed in the steps described in this section, we would like to highlight that this proposed distributed approach could be potentially extended to any other geometrical-based endmember extraction technique for performing linear spectral unmixing under the assumption of the presence of at least one pure pixel per endmember in the hyperspectral input data. The next section will introduce the main guidelines to extend the capabilities of the HyperCloud-RS framework by describing the mechanism for including new endmember extraction algorithms into the framework.

3.4 Guidelines for Integrating New EE Algorithms

This section presents the specifications to integrate new geometrical-based endmember extraction (EE) algorithms into the HyperCloud-RS framework and its proposed processing chain. In order to perform this integration, and considering that we used the Hadoop and Pig frameworks for instantiating the distributed architecture, it is required to:

- Embed the EE algorithm into a Pig User Defined Function (UDF); and
- Create its respective Pig Latin script.

In this sense, Algorithm 1 presents the general design to enclose a new endmember extraction algorithm into a Pig UDF, hereinafter referred to as EE-UDF. This structure will allow the instantiation of the particular EE technique in order to be adequately integrated within the HyperCloud-RS framework execution. In general, the EE-UDF takes for inputs:

- The URL for accessing the set of initial endmembers (allotted on a storage location in the cloud);
- The settings of the EE algorithm; and
- The hyperspectral subset, which is a data bag that contains the tuples to be analyzed with the EE algorithm.

Algorithm 1 Structure for designing the Endmember Extraction User Defined Function (EE-UDF)

1: Get the absolute path (URL) to the set of initial endmembers.
2: Provide the URL connection for stream reading.
3: Buffer the initial input data in local memory.
4: Get the options for the processing new EE algorithm.
5: Process the data from the hyperspectral subset with the new EE algorithm.
6: Return the candidate endmembers.

According to Algorithm 1, the set of initial endmembers is initially allocated in an auxiliary cloud repository, whose URL must be defined within the EE-UDF for performing the connection establishment and streaming the data to local memory within each processing node. This auxiliary repository is likewise used to store the promising endmembers, so they can be later accessed by each slave computing node in the cluster for performing the validation process of the promising endmembers.

Then, EE algorithm configurations are read and set, and the vertexes of the simplexes within each hyperspectral subset are computed, thus providing the set candidate endmembers, one by each slave node. Note that each subset is disjoint and is accordingly generated by the distributed framework.

Finally, the candidate endmembers are gathered by the master node, which, after processing them again with the EE algorithm, defines the set of promising endmembers. Those endmembers are then redistributed to the slave nodes for validating their consistency against each set of

candidate endmember within each slave node. In case the validation process fails, the EE process is repeated, but this time with the promising endmembers as the set of initial endmembers for the EE algorithm at each slave node; repeating this process up to the set of final endmembers is found, or a maximum number of iterations is completed.

The whole endmember extraction process must be encoded into a Pig Latin script, as presented in Algorithm 2. The script contains instructions for registering the EE-UDFs and all the libraries required. The EE algorithm and its particular parameters should be defined in the script, as well as the absolute path to the hyperspectral dataset and to the set of initial endmembers.

Algorithm 2 Pig Latin script for the Endmember Extraction Process definition

1: **REGISTER** the path to EE-UDF files.
2: **REGISTER** the path to Libraries files.
3: **DEFINE** the EE algorithm to be executed
4: Define the path to the set of initial endmembers.
5: Define the EE algorithm parameters.
6: **LOAD** the complete hyperspectral dataset.
7: **FOREACH** subset in the hyperspectral dataset **GENERATE** their candidate endmembers by executing the EE-UDF.
8: **REDUCE** the processing outcomes.
9: **GATHER** the candidate endmembers at the master node.
10: Perform the EE algorithm on the candidate endmembers to find the promising endmembers.
11: Distribute the promising endmembers and validate the outcome.
12: In case the promising endmembers are not stable, repeat from step 7.

Then as previously described, upon execution, the EE-UDF process each tuple in its own subset, providing the candidate endmembers, which represent the results of the distributed processes, and that are later merged at the master node in the reduction step. Finally, the candidate endmembers are used for creating the promising endmembers, and the validation process is executed.

Chapter IV

Experimental Design and Results

To assess the proposed distributed approach and its implementation, we conducted a set of experiments using the well-known Cuprite hyperspectral dataset. This section reports such experiments and the analysis carried out in this work.

4.1 Dataset

The AVIRIS (Airborne Visible Infra-Red Imaging Spectrometer, operated by NASA's Jet Propulsion Laboratory) Cuprite dataset was used in our experiments to evaluate the performance of our approach in extracting endmembers. The Cuprite scene [45] was collected over the Cuprite mining district in Nevada in the summer of 1997, and it contains 350×350 pixels with 20 m spatial resolution, 224 spectral bands in the range 0.4–$2.5\,\mu$m and a nominal spectral resolution of $10\,\eta$m, which are available in reflectance units after atmospheric correction, with a total data size of around 50 Mb.

Within the original dataset, spectral bands 1–6, 105–115, 150–170, 222–224 were removed prior to the analysis due to water absorption and low SNR, retaining 183 spectral bands. The Cuprite subset used in the experiments corresponds to the upper rightmost corner of the sector

labeled as f970619t01p02r02 and can be found online at http://aviris.jpl.nasa.gov/data/free_data.html. Figure 4.1 presents a false color composition of the Cuprite image used in the experiments, providing reference to its scale and orientation as well.

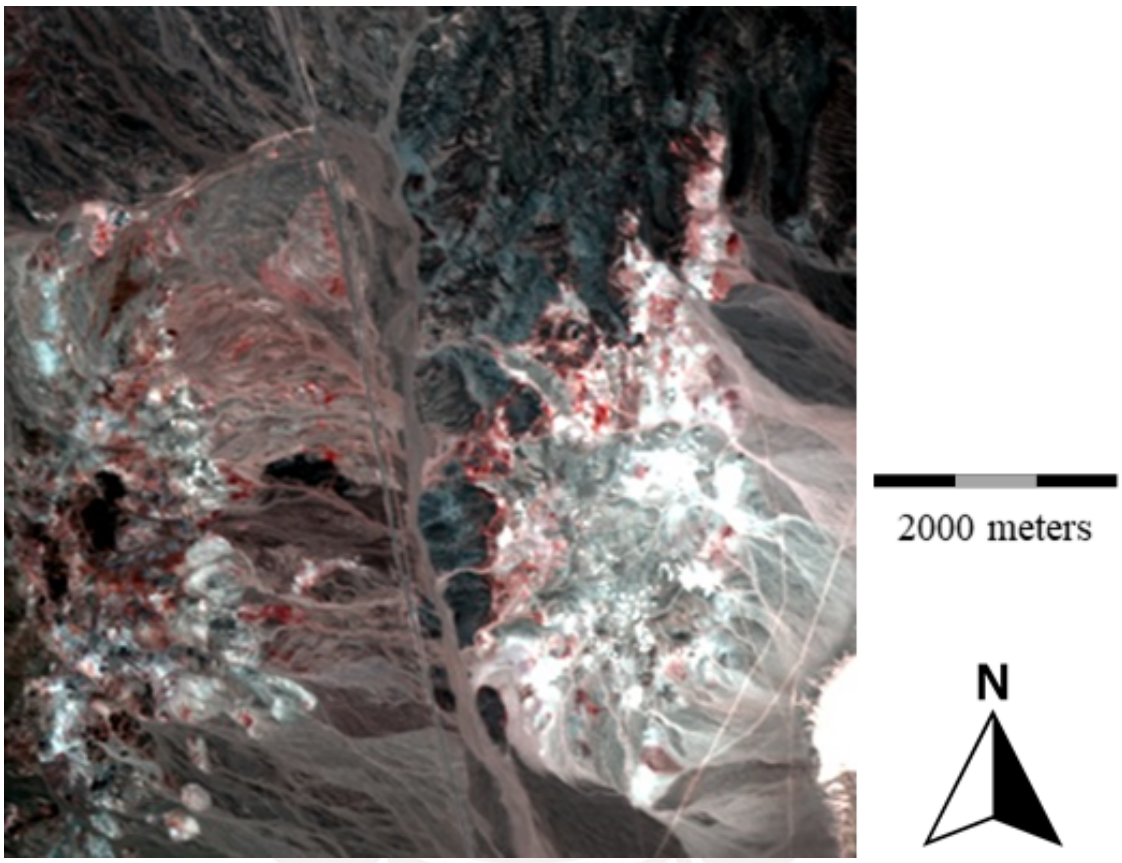

Figure 4.1: Cuprite hyperspectral image (False color composition, with the 33rd, 15th, and 11th spectral bands for the red, green and blue layers, respectively).

The Cuprite scene is well understood mineralogically and contains many reference ground signatures of the main minerals of interest. The scene encloses a total of 16 endmembers, five of which represent pure materials: Alunite, Buddingtonite, Calcite, Kaolinite, and Muscovite. The spectral signature of these minerals is available at the United States Geological Survey (USGS) library (available at http://speclab.cr.usgs.gov/spectral-lib.html). This library and its signatures were used in this work to assess the accuracy of the outcomes provided

by the endmember extraction technique, for both the sequential and the distributed version of the N-FINDR algorithm.

For estimating the number of endmembers in the Cuprite dataset we used the hyperspectral signal identification by minimum error (Hysime) algorithm [119] (we use its implementation available in [56]), providing a total of 16 endmembers in the scene. Then, we used the Principal Component Analysis algorithm to reduce the spectral dimensions of the image, retaining the first 15 principal components (in order to enable the construction of the 16-vertexes simplex), delivering an initial data size file of 30 Mb.

Based on this reduced dataset, three synthetic datasets were generated by replicating it $\times 100$, $\times 200$, and $\times 500$ times, producing approximately 3.1 Gb, 6.2 Gb and 15.1 Gb data size files, respectively. It is important to notice that these replications were made in the spatial dimension of the hyperspectral dataset, therefore the subsequent synthetic datasets represent image mosaics maintaining the same number of principal components as the reduced hyperspectral image, but with different image sizes.

The creation of these synthetic datasets is a required step for assessing the performance of our approach, as currently there is a general lack of large-scale public, controlled, and validated hyperspectral datasets. Actually, in the evaluation of related cloud-based methods as the ones in [44, 57, 78, 114, 111], similar synthetic data enlargement was performed as the one here described.

4.2 Cloud Environment

It is worth mentioning that although the cloud service chosen for conducting the experiments was Amazon Web Services (AWS) [120], the framework could be easily extended to integrate with any other cloud service (e.g. Google Cloud Platform [121], Microsoft Azure [122], IBM Bluemix [123]).

Thus, Amazon Simple Storage Service (S3) was used to store the UDFs, the Java libraries, the hyperspectral dataset, its synthetic versions, all the auxiliary data, and the libraries required for executing the distributed process. Amazon Elastic MapReduce (EMR) was used to manage the Hadoop framework, for distributing and processing the data across the nodes in the cluster, which was dynamically built using Amazon Elastic Compute Cloud (EC2) instances. Additionally, with Amazon EMR service is possible to select machines that have Hadoop and Pig already installed, thus the configurations of the machines could be effortlessly customized.

For the experiments, clusters with an increasing number of nodes were configured and used, starting with 2 nodes (baseline), and then scaling them with 4, 8, 16, and 32 nodes at each new configuration. The computer nodes available in AWS range from basic to high-performance machines; for the experiments, the m5.xlarge machine types were used, containing an Intel Xeon Platinum 8175M series processors operating at 2.5 GHz with 4 vCPUs, and 16 GB of RAM [124], and the Hadoop 2.10.1 and Pig 0.17.0 versions were configured as well.

Another important remark is that, although the machines were equipped with four virtual cores, the processing tasks were executed over a single core, recalling that the proposed distributed implementation was specifically designed to tackle the problem of processing very large volumes of hyperspectral remote sensing data, abstracting from particular hardware configurations.

Although the distributed endmember extraction process could be more efficient with the use of all the available cores in the computing nodes, this research was mostly concerned with the relative performance gains brought by scaling up homogeneous computer grids, more specifically, in terms of increasing the number of machines that compose those grids. We are aware of the potential computational efficiency advantages that could be achieved by jointly exploiting other programming models, such as the multicore-based ones, but that would not contribute to the analysis of our primary focus, that is, managing large volumes of hyperspectral remote

sensing data.

Another important characteristic of this architecture is that one of the nodes always acts as the master node, which is responsible for scheduling and managing the processing tasks through the Hadoop JobTracker, and which is not available for executing the target processing tasks. In this sense, in order to make a fair comparison among the sequential and distributed versions of our proposed N-FINDR implementation, we used the two-node cluster configuration to provide a feasible approximation of the sequential processing times for assessing each synthetic dataset. Furthermore, as the same distributed processing framework and file system (provided by Hadoop) are installed in this baseline configuration, we can ensure that the speedups eventually achieved by using larger clusters would be solely due to their scaling up, i.e., including additional machines.

All the experiments were performed using the implementation of the HyperCloud-RS Framework described in the previous chapter. The experimental results, presented in the following sections, represent the average of 10 executions of the combination of each synthetic dataset and the number of nodes in the cluster, and they are used for assessing the distributed N-FINDR algorithm in terms of both accuracy and computation performance when compared against its sequential baseline implementation.

4.3 Accuracy Assessment

Regarding the accuracy, we conducted a series of experiments to demonstrate the validity of our framework for extracting endmembers when working on large hyperspectral datasets. We compared the estimated endmembers, computed with our framework, against the ground-truth spectral signatures from the USGS library, available at: `https://crustal.usgs.gov/speclab/QueryAll07a.php`. For such comparison, we used the metric described in [94], which is defined as:

$$\phi_E = \frac{\left\| E - \widehat{E} \right\|_F}{\|E\|} \tag{4.1}$$

In Equation (4.1), $\|\cdot\|_F$ stands for the Frobenius norm, $\|\cdot\|$ is the Euclidean norm, $\widehat{E}$ represents the estimated endmember signatures, and E denote the ground-truth endmember signatures [94]. It is worth mentioning that endmember extraction algorithms return the most accurate results when ϕ_E tends to zero, which is the best possible value for that metric.

Following the common procedure used in the evaluation of endmember extraction methods, and before computing the similarity metric given by Equation (4.1), we performed a spectral feature matching between the endmembers delivered by our method and the spectral signatures references provided by the USGS library. The objective of that procedure is to identify the most similar ground-truth endmembers that correspond to the ones computed with our method.

Such signature matching is based on the spectral angle distance (SAD) metric, described in Equation (4.2), which compares the distance between two spectral vectors $\boldsymbol{e}_i$ and $\boldsymbol{e}_j$. The pair of endmembers associated with the lowest SAD values are then considered as corresponding endmembers. The SAD metric is defined as:

$$d_{SAD}\left(\boldsymbol{e}_i, \boldsymbol{e}_j\right) = \arccos \frac{\boldsymbol{e}_i \cdot \boldsymbol{e}_j}{\|\boldsymbol{e}_i\| \cdot \|\boldsymbol{e}_j\|} \tag{4.2}$$

where $\{\boldsymbol{e}_i\}_{i=1}^{N}$ represents the set of the spectral signatures in the USGS library, and $\{\boldsymbol{e}_j\}_{j=1}^{R}$ represents the estimated endmember signatures set, with N as the total number of spectral signatures in USGS library, and R as the total number of estimated endmembers.

Thus, for assessing the accuracy of our method in terms of the similarity metric given by Equation (4.1), we first used the original hyperspectral image of 350×350 pixels for finding the estimated endmember sets, for both, the sequential version of the N-FINDR algorithm (executed on a standalone machine) and its proposed distributed implementation, executed on

cluster environments with 2, 4, 8, 16, and 32 computing nodes. Then, following the procedure described in the last paragraph, and according to the Equation (4.2), we performed the spectral signature matching using the SAD distance for each cluster configuration set, hence defining the corresponding ground-truth endmember sets, which contain the closest sixteen spectral signatures from the USGS library. Table 4.1 presents the values of the ϕ_E metric for the sequential and distributed versions of the N-FINDR algorithm.

Table 4.1: Accuracy (ϕ_E) obtained with sequential processing of the N-FINDR algorithm, and with the proposed distributed version, over different cluster configurations on the Cuprite image.

Sequential	**Distributed N-FINDR Algorithm**				
N-FINDR	**02 Nodes**	**04 Nodes**	**08 Nodes**	**16 Nodes**	**32 Nodes**
0.0984	0.0984	0.0984	0.0984	0.0984	0.0984

From Table 4.1 it can be observed that the proposed distributed approach delivers the exact same accuracy results as the sequential implementation. Furthermore, all the ϕ_E metric values are close to zero (which represents the best possible value), assuring that the distributed implementation of the algorithm provides not only the same set of estimated endmembers, as its sequential counterpart and regardless of the particular cluster configuration used, but also does it with high precision.

4.3.1 Endmember Validation

The endmember extraction accuracy can be validated in terms of the quality of the reconstruction of the original Cuprite hyperspectral dataset. The reconstruction process of the original hyperspectral image is performed using the set of estimated endmembers (which with our approach are the same for the sequential and distributed executions of the N-FINDR algorithm, as previously stated), and their estimated fractional abundance maps, which can be computed by means of the Fully Constrained Linear Spectral Unmixing [125]. Then, we can obtain the recon-

structed image by combining the estimated endmember set and their correspondent estimated fractional abundance maps.

The reconstructed image may then be compared to the original Cuprite scene using the root-mean-square error (RMSE), defined in Equation (4.3) [44] as:

$$\text{RMSE} = \sqrt{\frac{1}{n \times L} \|\boldsymbol{X} - \hat{\boldsymbol{\alpha}}\boldsymbol{M}\|_F^2} \tag{4.3}$$

where L and n stand for the number of bands and pixels in the image $\boldsymbol{X}$ of size $n \times L$, respectively. $\hat{\boldsymbol{\alpha}}$ represents the estimated fractional abundances coefficient matrix of size $n \times p$, recalling that p is the number of endmembers in the image, and $\boldsymbol{M}$ is the estimated endmember matrix of size $p \times L$.

Lower RMSE scores correspond to a higher similarity between the compared images, and a set of high-quality endmembers and their associated estimated abundances can provide higher precision in the reconstruction of the original scene [65].

In Figure 4.2 we show per-pixel RMSE scores computed comparing the reconstructed image and the original one. As it can be observed, the RMSE errors are very low (considering a mean overall RMSE value equal to 2.65×10^{-5}), with and homogeneous spatial distribution, thus indicating an adequate overall reconstruction of the original image, and, therefore, an accurate estimation of endmembers provided by our method.

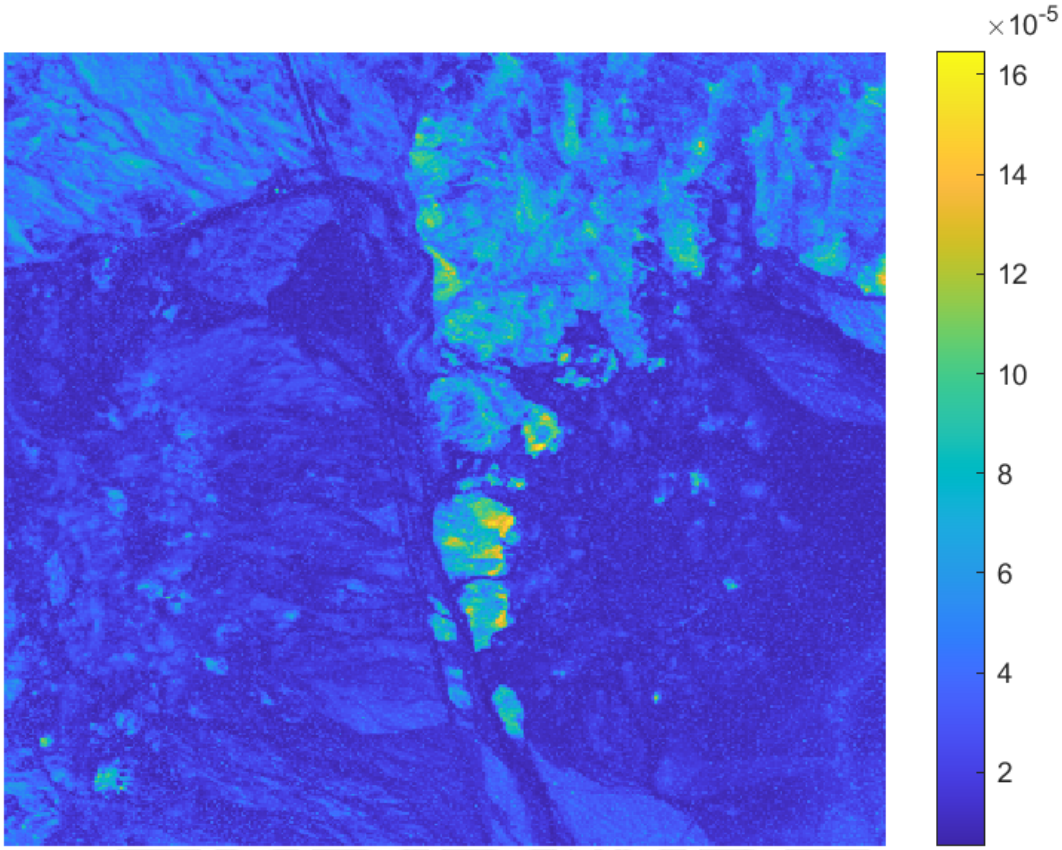

Figure 4.2: Per−pixel RMSE computed with the reconstruction of the Cuprite hyperspectral dataset.

4.4 Computational Performance Assessment

Regarding the assessment of computation performance, in Table 4.2 we present the average processing times for (from top to bottom):

- reading the data;
- processing the data (including the process of storing the outcomes on the cloud repository); and, finally,
- the entire length of time for completing the endmember extraction process.

Table 4.2: Average processing times for each step of the endmember extraction process on the cloud environment (in seconds).

Synthetic Dataset	02 Nodes	04 Nodes	08 Nodes	16 Nodes	32 Nodes	Processing Times
Cuprite	17.3	17.4	16.5	16.1	16.3	data transfer time
3.1 Gb	6531.6	2681.1	1973.3	1605.4	1352.0	data processing time
	6548.9	2698.5	1989.8	1621.5	1368.3	total time
Cuprite	29.4	29.1	28.9	28.3	28.5	data transfer time
6.2 Gb	12,433.9	4349.4	3517.4	1757.4	1409.3	data processing time
	12,463.3	4378.5	3546.3	1785.7	1437.8	total time
Cuprite	83.7	84.1	83.6	83.9	81.2	data transfer time
15.1 Gb	31,194.5	10,464.2	7812.7	3890.7	1897.0	data processing time
	31,278.2	10,548.3	7896.3	3974.6	1978.2	total time

As expected, observing the values in Table 4.2, the data processing time was the largest relative to the entire processing time, furthermore, from this table, it can likewise be seen that the times involved in reading the data do not vary substantially, whereas the time consumed by the endmember extraction process quickly decreases as more nodes are used in the cluster.

For further assessing the computation performance gains achieved by increasing the number of cluster nodes, we computed the speedups, based on the complete processing times, achieved with our distributed approach running on clusters with 2, 4, 8, 16, and 32 nodes, as presented in Table 4.3. Complementary, Figure 4.3 shows the speedups achieved with the 4, 8, 16, and 32 node configurations, on the enlarged versions of the Cuprite dataset, in relation to the processing time related to the 2 node configuration, which actually represents the execution of the approach by using only one processing node, as explained in Section 4.2.

Table 4.3: Speedups for the distributed approach of the N-FINDR algorithm on the Cuprite synthetic datasets.

Synthetic Dataset	02 Nodes	04 Nodes	08 Nodes	16 Nodes	32 Nodes
Cuprite 3.1 Gb	1.0	2.43	3.29	4.04	4.79
Cuprite 6.2 Gb	1.0	2.85	3.51	6.98	8.67
Cuprite 15.1 Gb	1.0	2.97	3.96	7.87	15.81

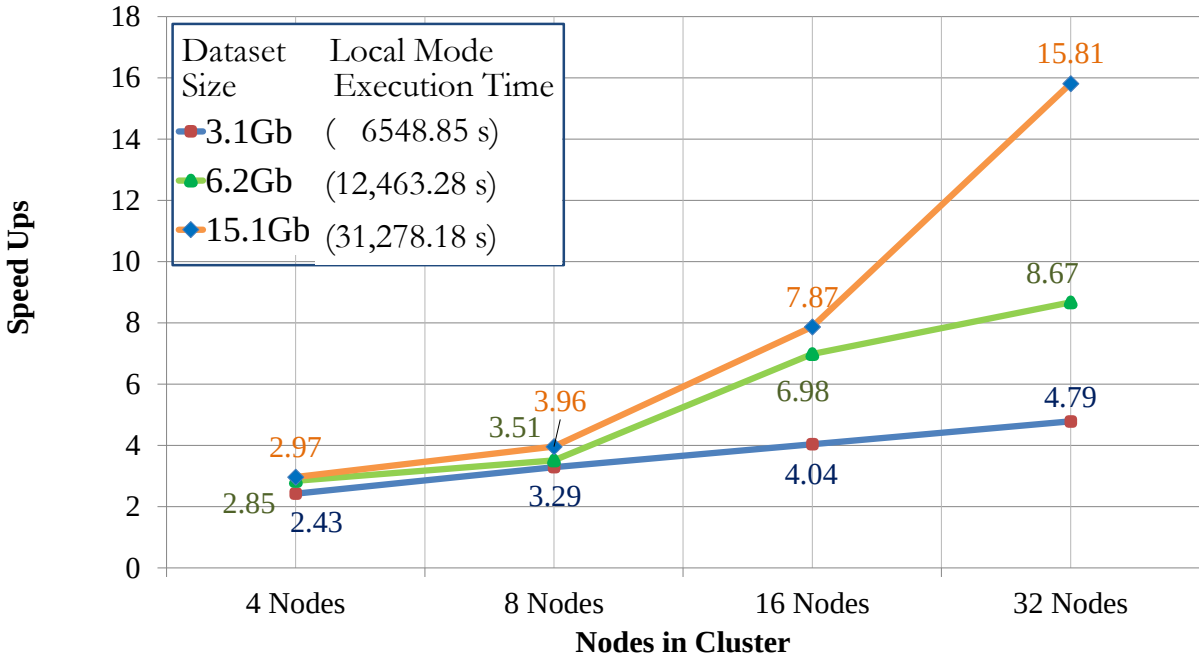

Figure 4.3: Speedups for the distributed approach of the N-FINDR algorithm on the Cuprite synthetic datasets.

Regarding the values shown in Table 4.3, considering the first synthetic dataset (3.1 Gb), the speedups were 2.43, 3.29, 4.04, and 4.79, for 4, 8, 16, and 32 nodes, respectively. The attained

speedups indicate that as the number of nodes increases, each cluster configuration delivers higher speedups, as expected. Furthermore, it can be noticed that this behavior is retained among the other synthetic datasets, but at different growth rates, which are mainly guided by the dataset size to be processed.

However, and according to Figure 4.3, it can be observed that smaller data volumes imply a lower scalability potential, whereas bigger data volumes allow for achieving higher speedups, which is directly related to proper exploitation of the distributed environment, where the larger the size of the data to be distributed and processed, the better the performance that could be achieved.

Also referring to Figure 4.3, the speedups showed an almost linear growth when 4 and 8 nodes were used, regardless of the dataset size. Notwithstanding, as we increased the number of nodes in the cluster, the speedups were likewise improved, but such improvement was remarkably better for the largest dataset size, e.g., 15.1 Gb.

Thus, for example, with 32 nodes the speedup ranged from 4.79 to 15.81 as the size of the synthetic dataset increased from 3.1 Gb to 15.1 Gb. Those results confirm our assumption that smaller dataset sizes result in lower speedup values and, as the dataset size is increased, the speedup also increases.

Chapter V

Discussion

The improvements in hyperspectral remote sensing systems, considering their spatial and spectral resolutions and the increasing rates of information produced by hyperspectral sensors, impose significant challenges related to adequately storing and efficiently processing large volumes of image data [26, 14, 16, 27, 25, 32, 29, 9].

In this regard, high-performance computing systems have emerged as potential solutions to face those challenges. Such solutions include approaches based on multicore processing [64], GPUs [75, 40, 66], FPGAs [67, 68], and computer clusters [3, 70]. Although many methods based on the approaches just mentioned have proven high efficiency in terms of processing speed, they still struggle to adequately manage large-scale data problems, mainly due to their limited memory capacity [77].

More recently, cloud computing-based systems have emerged as feasible alternatives to handle data-intensive applications, as described in [57, 48, 58, 78, 59, 38]. However, there are still a number of issues to be considered and investigated in the design of cloud-based solutions for remote sensing problems [34, 47, 48, 4], particularly with respect to the implementation of distributed unmixing algorithms, which are highly complex and computationally demanding processes [44].

A remarkable example in that context is the work presented in [44], in which the authors implemented a parallel unmixing-based content retrieval built on top of a cloud computing platform. That work introduced a distributed version of the Pixel Purity Index (PPI) algorithm for the endmember extraction process, which, as the N-FINDR algorithm, belongs to the same class of pure pixel geometrical-based approaches for performing linear spectral unmixing. Regarding their experiments, a 22.40 Gb dataset (re-scaled from the original Cuprite image, also used in this work) required a total processing time of 5400 seconds, when using a 32-node cluster configuration, for finding the endmembers in the scaled version of the Cuprite dataset.

Analogously, we observe that in our approach, we required only 1978.2 seconds to process a similar dataset, properly dealing with the limitations the authors of [44] describe as: "the parallel strategy for unmixing algorithms should be well designed"', and confirming that "unmixing algorithms are selectable for higher computing speed". Then, and based on our experiments and results, such issues could be largely covered with the implementation of our framework, further considering that this work is open to the inclusion of potentially any geometrical-based algorithm for endmember extraction.

Regarding the computation performance, the results presented in Figure 4.3 indicate that the speedups achieved with our implementation described a linear trend for the lower nodes configuration, regardless of the dataset size being processed, which is also in line with the endmember extraction performance described in [44], where authors also experienced linear growths as the number of nodes is less than a certain quantity, as we pointed out previously. Furthermore, as depicted in that figure, the smaller the dataset size, the lower the acceleration gains, thus implying a diminished scalability potential; on the other hand, the results also show that as larger volumes of data are processed, higher speedups can be achieved, but again, up to a certain point, thus defining somehow an implicit trade-off between the dataset sizes to be processed and the number of nodes to be configured in the cluster.

To better describe the implications of such a trade-off, in Table 5.1 we present the proportional speedup increments, computed considering the ratio between speedup achieved with our distributed approach and the number of nodes in the cluster. The table shows that those ratios are significantly higher for the largest dataset; for instance, considering the cluster configurations with 4, 8, 16, and 32 nodes, such ratios are 0.74, 0.50, 0.49, and 0,49, for the 15.1 Gb dataset; and 0.61, 0.41, 0.25, and 0.15 for the 3.1 Gb dataset. Those results are actually very interesting as they demonstrate the scalability limits of our approach.

Table 5.1: Proportional speedup increments for each node configuration on the cloud environment.

Synthetic Dataset	**04 Nodes**	**08 Nodes**	**16 Nodes**	**32 Nodes**
Cuprite 3.1 Gb	0.61	0.41	0.25	0.15
Cuprite 6.2 Gb	0.71	0.44	0.44	0.27
Cuprite 15.1 Gb	0.74	0.50	0.49	0.49

Extending this analysis, taking into account the smaller dataset (3.1 Gb) for example, increasing the number of cluster nodes decreases the efficiency of the method (regarding "efficiency" as the proportion of the theoretical maximum speedup obtainable for a given number of nodes). On the other side, considering the largest dataset (15.1 Gb), however, efficiency is maintained when increasing the number of cluster nodes.

In this sense, it is likewise interesting to observe in Table 5.1 and in Figure 4.3 the different performance of the speedup curves concerning the 6.2 Gb and 15.1 Gb datasets for the 16 and 32 node cluster configurations, in which the proportional increase in the speedup times is larger for the 32 node configuration. We then note that the distributed framework allocates fixed/limited memory space for each processing task, and distributes those tasks throughout the cluster nodes.

Then, if there are many tasks for the same node, the total processing time for that node will be higher than if the node had fewer tasks to process. Conversely, with fewer tasks per node,

processing time lowers, favoring speedups. However, that behavior is not expected to occur indefinitely with the increase in the number of computing nodes. At some point, adding nodes would not lead to any speedup increment, in fact, we expect that the opposite happens, i.e., speedups start to decrease because of the increased communication overhead, behavior which is also similar to what authors found and describe in [44], where they realized that speedups will not increase linearly as the number of cores increases.

Moreover, after some point, for a fixed dataset size, the speedup growth becomes slower, or even negative, when using a higher number of nodes. Indeed, and examining again the values in Table 5.1, the proportional decrease in the ratio between dataset size and computing nodes observed for the 6.2 Gb dataset from 16 to 32 nodes seems to be evidence of that issue, where we are probably using many more nodes in the cluster than the required to process a not so large volume of information.

Additionally, we must remark that in this work we have focused on describing the proposed distributed implementation of the N-FINDR algorithm on the HyperCloud-RS framework, and furthermore on providing guidelines on how to integrate new endmember extraction algorithms into the framework. Thus, and in contrast to related approaches described in [57, 78, 44, 114, 111], our framework provides the means to seamlessly implement other distributed endmember extraction algorithms on cloud computing infrastructures. We further and thoroughly believe that such capabilities overture a wide range of applications based on hyperspectral unmixing analysis.

We are aware that we did not report on the monetary costs involved and we did not discuss the trade-off between the efficiency and the cost of running our solution on the commercial AWS cloud-computing infrastructure services, as we believe that theme goes beyond the scope of this work, and they are discussed in publications specifically focused on that subject, such as the one presented in [126].

Nevertheless, a related topic that would be of great value for operational decisions concerning dealing with commercial cloud infrastructure services, is the development of tools that suggest alternative cluster configurations, considering not only dataset sizes but also time and monetary constraints for running distributed solutions such as the one described in this research. Once again, we believe that the development of such tools goes beyond the scope of this work, however, such analysis would be another interesting line for future research.

Finally, considering our particular implementation of the N-FINDR algorithm, the accuracy and computing performance observed in the experimental analysis demonstrate that our approach is capable of adequately managing large amounts of hyperspectral remote sensing data, thus representing a reliable and efficient solution for the endmember extraction process.

Specifically, regarding the computation performance, our distributed N-FINDR implementation outperformed a state-of-the-art, cloud-based distributed method for endmember extraction [44], and can, therefore, be used as a baseline for future research in the field.

Moreover, we demonstrated that the proposed HyperCloud-RS framework can be easily extended with the inclusion of other pure pixel geometrical-based approaches for linear spectral unmixing, thus enabling other researchers to easily implement and execute their own distributed approaches over cloud computing infrastructures.

Chapter VI

Conclusions

Current improvements in hyperspectral remote sensing systems establish compelling challenges for storing, management, deployment, processing, analysis, and interpretation of the large volumes of hyperspectral remote sensing data that such systems are currently providing, considering that hyperspectral image processing is a costly and complex computational process, whose analysis demands for efficient and scalable computing solutions, imposing significant requirements in terms of storage, processing, and near real-time responses.

To overcome the aforementioned processing issues, several specialized high-performance computing (HPC) systems have been proposed, from multicore-based approaches, up to systems based on graphics processing units (GPUs), field-programmable gate arrays (FPGAs), and computer clusters.

Nevertheless, despite the compelling capabilities provided by HPC systems, there are still important issues to be consigned for adequately dealing with large volumes of hyperspectral remote sensing data, which more recently are being interestingly addressed by solutions based on cloud computing systems, as these platforms provide flexible and scalable hardware resources, delivering applications and software as services (SaaS), as well as infrastructure and platform as a service, thus providing the opportunity of accessing infinite computing resources,

but currently there is still a limited number of efforts to date, and not enough viable solutions for adequately exploiting cloud computing infrastructures for large-scale hyperspectral image processing.

In this context, in this thesis, we implemented and validated what we defined as the HyperCloud -RS framework, which is a platform that enables, and adequately exploits, the use of cloud computing environments for elastically allocate processing power and storage space for effectively performing endmember extraction processes on large-scale hyperspectral remote sensing data. Furthermore, we introduced, and validated, a novel distributed version of the sequential N-FINDR endmember extraction algorithm, built on top of the proposed framework, able to adequately handle the hyperspectral data distribution and perform its execution in cloud computing environments, in a reliable, scalable, and efficient way; additionally supporting distributed execution, network communication, and fault tolerance, transparently and efficiently to the user, thus enabling efficient use of available computational resources.

As a further contribution of this work, we provided the main guidelines on how to extend the HyperCloud-RS framework capabilities with the addition of new endmember extraction algorithms, as long as these algorithms belong to the class of pure pixel geometrical-based approaches for performing linear spectral unmixing, in which case, their integration with this framework becomes a straightforward process, as described in Section 3.4.

The experimental analysis, which assessed the accuracy and computation performance of the proposed solution, demonstrated the scalability provided by the framework and its potential to handle large-scale hyperspectral datasets. Remarkably, higher speedups were achieved when the amount of data being processed was largely increased, that is, as the dataset size increased, clusters containing more nodes delivered higher speedups, thus better exploiting the distributed resources.

The results additionally showed that arbitrarily increasing the number of cluster nodes while

fixing the dataset size does not necessarily deliver a proportional reduction of the execution times of the distributed N-FINDR algorithm. Therefore, to optimize computational performance, there must be an adequate balance between the amount of data to be processed and the number of nodes to be used. That seems to indicate that the optimum cluster settings depend not only on the endmember extraction algorithm but also on the amount of hyperspectral data to be processed.

Regarding our particular approach for distributing the N-FINDR geometrical-based method for linear spectral unmixing, it has been observed that if the initial seeds, distributed among the cluster nodes, are the same at each execution, and the endmember extraction algorithm parameter values remain unchanged, the outcome of its distributed implementation is identical to that of the sequential version, regardless of the number of cluster nodes configured. Actually, we have forecasted such behavior from the beginning of the research, i.e., producing the same outcome as the sequential execution of the algorithm was a design requirement, which allows us to mainly focus on the implementation, assessment, and validation of the proposed distribution strategy of the N-FINDR algorithm.

6.1 Future Works

We believe that this work overtures the possibility of raising multiple further research, outset from the integration of a dimensionality reduction process into the framework, up to the possibility of testing and comparing the performance of many other endmember extraction algorithms.

Furthermore, interesting research to explore is to assess the extension of our endmember extraction approach, with the increasing of a multicore parallelized approach implementation of the algorithm, capable to perform its execution within the computing nodes in the cloud environment to be deployed.

Additionally, further experiments could be performed to investigate the effects of varying the endmember extraction algorithm settings with respect to the accuracy obtained with its distributed implementation.

Finally, considering the evolution and availability of cloud-computing infrastructure-as-a-service technologies, further research should be directed to investigate in detail the trade-off between the efficiency and the associated cost of using such services, as compared to the acquisition of the necessary infrastructure for implementing distributed algorithmic solutions such as the one proposed in this work.

Bibliography

[1] W. M. IBM, "10 key marketing trends for 2017 and ideas for exceeding customer expectations." Available online: https://bizibl.com/marketing/download/10-key-marketing-trends-2017-and-ideas-exceeding-customer-expectations, 2017. (accessed on: 2022-06-03).

[2] J. Li, Z. Liu, X. Lei, and L. Wang, "Distributed fusion of heterogeneous remote sensing and social media data: A review and new developments," *Proceedings of the IEEE*, vol. 109, no. 8, pp. 1350–1363, 2021.

[3] M. K. Pektürk and M. Ünal, "Performance-aware high-performance computing for remote sensing big data analytics," in *Data Mining* (C. Thomas, ed.), ch. 5, Rijeka: IntechOpen, 2018.

[4] M. Chi, A. Plaza, J. A. Benediktsson, Z. Sun, J. Shen, and Y. Zhu, "Big data for remote sensing: Challenges and opportunities," *Proceedings of the IEEE*, vol. 104, no. 11, pp. 2207–2219, 2016.

[5] J. Li, J. A. Benediktsson, B. Zhang, T. Yang, and A. Plaza, "Spatial technology and social media in remote sensing: A survey," *Proceedings of the IEEE*, vol. 105, no. 10, pp. 1855–1864, 2017.

[6] W. Boulila, M. Sellami, M. Driss, M. Al-Sarem, M. Safaei, and F. A. Ghaleb, "Rs-dcnn: A novel distributed convolutional-neural-networks based-approach for big remote-sensing image classification," *Computers and Electronics in Agriculture*, vol. 182, p. 106014, 2021.

[7] Y. Cheng, K. Zhou, J. Wang, and J. Yan, "Big earth observation data integration in remote sensing based on a distributed spatial framework," *Remote Sensing*, vol. 12, no. 6, 2020.

[8] L. Wang, Y. Ma, J. Yan, V. Chang, and A. Y. Zomaya, "pipscloud: High performance cloud computing for remote sensing big data management and processing," *Future Generation Computer Systems*, vol. 78, pp. 353–368, 2018.

[9] Y. Ma, H. Wu, L. Wang, B. Huang, R. Ranjan, A. Zomaya, and W. Jie, "Remote sensing big data computing: Challenges and opportunities," *Future Gener. Comput. Syst.*, vol. 51,

pp. 47–60, 2015. Special Section: A Note on New Trends in Data-Aware Scheduling and Resource Provisioning in Modern HPC Systems.

[10] P. Gamba, P. Du, C. Juergens, and D. Maktav, "Foreword to the special issue on "human settlements: A global remote sensing challenge"," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 4, no. 1, pp. 5–7, 2011.

[11] M. Chi, A. J. Plaza, J. A. Benediktsson, B. Zhang, and B. Huang, "Foreword to the special issue on big data in remote sensing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 10, pp. 4607–4609, 2015.

[12] B.-E. Boudriki and F. Freitag, "Sat-hadoop-processor: A distributed remote sensing big data processing software for earth observation applications," *Applied Sciences*, vol. 11, no. 22, 2021.

[13] J. M. Bioucas-Dias, A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot, "Hyperspectral remote sensing data analysis and future challenges," *IEEE Geoscience and Remote Sensing Magazine*, vol. 1, no. 2, pp. 6–36, 2013.

[14] J. A. Benediktsson and Z. Wu, "Distributed computing for remotely sensed data processing [scanning the section]," *Proceedings of the IEEE*, vol. 109, no. 8, pp. 1278–1281, 2021.

[15] G. Cavallaro, M. Riedel, M. Richerzhagen, J. A. Benediktsson, and A. Plaza, "On understanding big data impacts in remotely sensed image classification using support vector machine methods," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 10, pp. 4634–4646, 2015.

[16] Y. Hajjaji, W. Boulila, and I. R. Farah, "An improved tile-based scalable distributed management model of massive high-resolution satellite images," *Procedia Computer Science*, vol. 192, pp. 2931–2942, 2021.

[17] M. M. U. Rathore, A. Paul, A. Ahmad, B.-W. Chen, B. Huang, and W. Ji, "Real-time big data analytical architecture for remote sensing application," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 8, no. 10, pp. 4610–4621, 2015.

[18] M. Datcu, "Hd-03: Big data from earth observation: analytics, mining, semantics," in *Proceedings of the 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, July 2015.

[19] L. Zhang, Q. Du, and M. Datcu, "Special section guest editorial: Management and analytics of remotely sensed big data," *J. Appl. Remote Sens.*, vol. 9, no. 1, 2015.

[20] T. E. S. Agency, "Sentinel-1." Available online: `https://sentinels.copernicus.eu/web/sentinel/missions/sentinel-1`, 2022. (accessed on: 2022-05-03).

[21] O. Grabak, "Sentinel-1 mission status," in *Proceedings of the 15th Meeting of the International Ice Charting Working Group*, 2014.

[22] Copernicus, "Access to data." Available online: `https://www.copernicus.eu/en/access-data`, 2022. (accessed on: 2022-05-03).

[23] R. Knowelden and A. Grazia, *Copernicus Sentinel Data Access - 2019 Annual Report*. Copernicus and European Space Agency, 2020.

[24] E. O. S. Data and I. S. (EOSDIS), "Eosdis annual metrics reports." Available online: `https://earthdata.nasa.gov/eosdis/system-performance/eosdis-annual-metrics-reports`, 2020. (accessed on: 2022-05-03).

[25] L. Wanchoo, D. Kafle, and J. Soon, *EOSDIS FY2020 - Annual Metrics Report*. NASA, 2020.

[26] J. Guo, C. Huang, and J. Hou, "A scalable computing resources system for remote sensing big data processing using geopyspark based on spark on k8s," *Remote Sensing*, vol. 14, no. 3, 2022.

[27] M. Amani, A. Ghorbanian, S. A. Ahmadi, M. Kakooei, A. Moghimi, S. M. Mirmazloumi, S. H. A. Moghaddam, S. Mahdavi, M. Ghahremanloo, S. Parsian, Q. Wu, and B. Brisco, "Google earth engine cloud computing platform for remote sensing big data applications: A comprehensive review," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 13, pp. 5326–5350, 2020.

[28] E. Erlingsson, G. Cavallaro, M. Riedel, and H. Neukirchen, "Scaling support vector machines towards exascale computing for classification of large-scale high-resolution remote sensing images," in *2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 1792–1795, 2018.

[29] J.-G. Lee and M. Kang, "Geospatial big data: Challenges and opportunities," *Big Data Research*, vol. 2, no. 2, pp. 74–81, 2015. Visions on Big Data.

[30] D. Kishor, "Big data: The new challenges in data mining," *Int. j. innov. res. comput. sci. technol.*, vol. 1, no. 2, pp. 39–42, 2013.

[31] M. Riedel, G. Cavallaro, and J. A. Benediktsson, "Practice and experience in using parallel and scalable machine learning in remote sensing from hpc over cloud to quantum computing," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pp. 1571–1574, 2021.

[32] B. Zhang, Z. Chen, D. Peng, J. A. Benediktsson, B. Liu, L. Zou, J. Li, and A. Plaza, "Remotely sensed big data: evolution in model development for information extraction [point of view]," *Proceedings of the IEEE*, vol. 107, no. 12, pp. 2294–2301, 2019.

[33] Y. Liu, L. Dang, S. Li, K. Cai, and X. Zuo, "Research progress on models, algorithms, and systems for remote sensing spatial-temporal big data processing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 5918–5931, 2021.

[34] Z. Wu, J. Sun, Y. Zhang, Z. Wei, and J. Chanussot, "Recent developments in parallel and distributed computing for remotely sensed big data processing," *Proceedings of the IEEE*, vol. 109, no. 8, pp. 1282–1305, 2021.

[35] X. Yao, G. Li, J. Xia, J. Ben, Q. Cao, L. Zhao, Y. Ma, L. Zhang, and D. Zhu, "Enabling the big earth observation data via cloud computing and dggs: Opportunities and challenges," *Remote Sensing*, vol. 12, no. 1, 2020.

[36] X. Huang, L. Wang, J. Yan, Z. Deng, S. Wang, and Y. Ma, "Towards building a distributed data management architecture to integrate multi-sources remote sensing big data," in *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*, pp. 83–90, 2018.

[37] W. Boulila, I. R. Farah, and A. Hussain, "A novel decision support system for the interpretation of remote sensing big data," *Earth Science Informatics*, vol. 11, no. 1, pp. 31–45, 2018.

[38] V. A. Ayma, G. A. O. P. da Costa, P. N. Happ, R. Q. Feitosa, R. d. S. Ferreira, D. A. B. Oliveira, and A. Plaza, "A new cloud computing architecture for the classification of remote sensing data," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 10, no. 2, pp. 409–416, 2017.

[39] S. Schade, "Big data breaking barriers - first steps on a long trail," *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci. - ISPRS Arch.*, vol. XL-7/W3, pp. 691–697, 2015.

[40] A. Yusuf and S. Alawneh, "A survey of gpu implementations for hyperspectral image classification in remote sensing," *Canadian Journal of Remote Sensing*, vol. 44, no. 5, pp. 532–550, 2019.

[41] G. Camps-Valls, D. Tuia, L. Gómez-Chova, S. Jiménez, and J. Malo, *Remote Sensing Image Processing*. Morgan & Claypool Publishers, 1 ed., 2011.

[42] J. Richards, *Remote Sensing Digital Image Analysis*. Springer Berlin, Heidelberg, 5 ed., 2013.

[43] A. Goetz, G. Vane, J. Solomon, and B. Rock, "Imaging spectrometry for earth remote sensing," *Science*, vol. 228, no. 4704, pp. 1147–1153, 1985.

[44] P. Zheng, Z. Wu, J. Sun, Y. Zhang, Y. Zhu, Y. Shen, J. Yang, Z. Wei, and A. Plaza, "A parallel unmixing-based content retrieval system for distributed hyperspectral imagery repository on cloud computing platforms," *Remote Sensing*, vol. 13, no. 2, pp. 1–21, 2021.

[45] Jet Propulsion Laboratory - California Institute of Technology, "Aviris data - ordering free aviris standard data products." Available online: `https://aviris.jpl.nasa.gov/data/free_data.html`, April 2015. (accessed on: 2022-06-03).

[46] S. Zhang, Y. Su, X. Xu, J. Li, C. Deng, and A. Plaza, *Recent Advances in Hyperspectral Unmixing Using Sparse Techniques and Deep Learning*, pp. 377–405. Springer International Publishing, April 2020.

[47] P. Ghamisi, N. Yokoya, J. Li, W. Liao, S. Liu, J. Plaza, B. Rasti, and A. Plaza, "Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art," *IEEE Geoscience and Remote Sensing Magazine*, vol. 5, no. 4, pp. 37–78, 2017.

[48] J. M. Haut, M. E. Paoletti, S. Moreno-Álvarez, J. Plaza, J.-A. Rico-Gallego, and A. Plaza, "Distributed deep learning for remote sensing data interpretation," *Proceedings of the IEEE*, vol. 109, no. 8, pp. 1320–1349, 2021.

[49] Q. Du, B. Tang, W. Xie, and W. Li, "Parallel and distributed computing for anomaly detection from hyperspectral remote sensing imagery," *Proceedings of the IEEE*, vol. 109, no. 8, pp. 1306–1319, 2021.

[50] M. E. Paoletti, J. M. Haut, J. Plaza, and A. Plaza, "Scalable recurrent neural network for hyperspectral image classification," *The Journal of Supercomputing*, vol. 76, pp. 8866–8882, 2020.

[51] Z. Wu, Y. Li, A. Plaza, J. Li, F. Xiao, and Z. Wei, "Parallel and distributed dimensionality reduction of hyperspectral data on cloud computing architectures," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 9, no. 6, pp. 2270–2278, 2016.

[52] A. Fontanella, E. Marenzi, E. Torti, G. Danese, A. Plaza, and F. Leporati, "A suite of parallel algorithms for efficient band selection from hyperspectral images," *J. Real-Time Image Process.*, vol. 15, no. 3, p. 537–553, 2018.

[53] E. Torti, G. Danese, F. Leporati, and A. Plaza, "A hybrid cpu–gpu real-time hyperspectral unmixing chain," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 9, no. 2, pp. 945–951, 2016.

[54] T. Hey, S. Tansley, and K. Tolle, *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, October 2009.

[55] A. Marinoni and P. Gamba, "Nonlinear endmember extraction in earth observations and astroinformatics data interpretation and compression," in *2015 Int. Geosci. Remote Sens. Symp. (IGARSS)*, pp. 1500–1503, 2015.

[56] J. M. Bioucas-Dias, A. Plaza, N. Dobigeon, M. Parente, Q. Du, P. Gader, and J. Chanussot, "Hyperspectral unmixing overview: Geometrical, statistical, and sparse regression-based approaches," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 5, no. 2, pp. 354–379, 2012.

[57] Z. Wu, J. Sun, Y. Zhang, Y. Zhu, J. Li, A. Plaza, J. A. Benediktsson, and Z. Wei, "Scheduling-guided automatic processing of massive hyperspectral image classification on cloud computing architectures," *IEEE Trans. Cybern.*, vol. 51, no. 7, pp. 3588–3601, 2021.

[58] J. M. Haut, J. A. Gallardo, M. E. Paoletti, G. Cavallaro, J. Plaza, A. Plaza, and M. Riedel, "Cloud deep networks for hyperspectral image analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 12, pp. 9832–9848, 2019.

[59] J. Sun, Y. Zhang, Z. Wu, Y. Zhu, X. Yin, Z. Ding, Z. Wei, J. Plaza, and A. Plaza, "An efficient and scalable framework for processing remotely sensed big data in cloud computing environments," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 7, pp. 4294–4308, 2019.

[60] M. A. Veganzones, J. E. Cohen, R. Cabral Farias, J. Chanussot, and P. Comon, "Nonnegative tensor cp decomposition of hyperspectral data," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 5, pp. 2577–2588, 2016.

[61] L. Yang, X. Sun, L. Peng, X. Yao, and T. Chi, "An agent-based artificial bee colony (abc) algorithm for hyperspectral image endmember extraction in parallel," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 8, no. 10, pp. 4657–4664, 2015.

[62] D. M. S. Arsa, G. Jati, A. J. Mantau, and I. Wasito, "Dimensionality reduction using deep belief network in big data case study: Hyperspectral image classification," in *2016 International Workshop on Big Data and Information Security (IWBIS)*, pp. 71–76, 2016.

[63] G. Cavallaro, D. B. Heras, Z. Wu, M. Maskey, S. López, P. Gawron, M. Coca, and M. Datcu, "High-performance and disruptive computing in remote sensing: Hdcrs-a new working group of the grss earth science informatics technical committee [technical committees]," *IEEE Geoscience and Remote Sensing Magazine*, vol. 10, no. 2, pp. 329–345, 2022.

[64] S. Bernabé, L. I. Jiménez, C. García, J. Plaza, and A. Plaza, "Multicore real-time implementation of a full hyperspectral unmixing chain," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 5, pp. 744–748, 2018.

[65] S. Bernabé, G. Botella, G. Martín, M. Prieto-Matias, and A. Plaza, "Parallel implementation of a full hyperspectral unmixing chain using opencl," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 10, no. 6, pp. 2452–2461, 2017.

[66] L. I. Jiménez, S. Sánchez, G. Martín, J. Plaza, and A. J. Plaza, "Parallel implementation of spatial–spectral endmember extraction on graphic processing units," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 10, no. 4, pp. 1247–1255, 2017.

[67] M. Díaz, R. Guerra, S. López, J. Caba, and J. Barba, "An fpga-based implementation of a hyperspectral anomaly detection algorithm for real-time applications," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, pp. 1579–1582, 2021.

[68] C. Li, L. Gao, A. Plaza, and B. Zhang, "Fpga implementation of a maximum simplex volume algorithm for endmember extraction from remotely sensed hyperspectral images," *J. Real Time Image Process.*, vol. 16, pp. 1681–1694, 2019.

[69] C. Gonzalez, D. Mozos, J. Resano, and A. Plaza, "Fpga implementation of the n-findr algorithm for remotely sensed hyperspectral image analysis," *IEEE Trans. Geosci. Remote Sens.*, vol. 50, no. 2, pp. 374–388, 2012.

[70] S. Sánchez, A. Paz, G. Martín, and A. Plaza, "Parallel unmixing of remotely sensed hyperspectral images on commodity graphics processing units," *Concurrency and Computation: Practice and Experience*, vol. 23, no. 13, pp. 1538–1557, 2011.

[71] T. G. Cervero, J. Caba, S. López, J. D. Dondo, R. Sarmiento, F. Rincón, and J. López, "A scalable and dynamically reconfigurable fpga-based embedded system for real-time hyperspectral unmixing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 8, no. 6, pp. 2894–2911, 2015.

[72] S. Mittal and J. S. Vetter, "A survey of methods for analyzing and improving gpu energy efficiency," *ACM Computing Surveys*, vol. 47, no. 2, pp. 1–23, 2015.

[73] A. Plaza, Q. Du, Y.-L. Chang, and R. L. King, "High performance computing for hyperspectral remote sensing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 4, no. 3, pp. 528–544, 2011.

[74] E. Erlingsson, G. Cavallaro, H. Neukirchen, and M. Riedel, "Scalable workflows for remote sensing data processing with the deep-est modular supercomputing architecture," in *2019 IEEE International Geoscience and Remote Sensing Symposium*, pp. 5905–5908, 2019.

[75] Z. Wu, J. Liu, S. Ye, L. Sun, and Z. Wei, "Optimization of minimum volume constrained hyperspectral image unmixing on cpu–gpu heterogeneous platform," *Journal of Real-Time Image Processing*, vol. 15, pp. 265–277, 2018.

[76] A. Plaza, D. Valencia, and J. Plaza, "An experimental comparison of parallel algorithms for hyperspectral analysis using heterogeneous and homogeneous networks of workstations," *Parallel Computing*, vol. 34, no. 2, pp. 92–114, 2008.

[77] Z. Wu, S. Ye, J. Wei, Z. Wei, L. Sun, and J. Liu, "Fast endmember extraction for massive hyperspectral sensor data on gpus," *Int. J. Distrib. Sens. Netw.*, vol. 9, no. 10, pp. 1–7, 2013.

[78] J. Sun, H. Li, Y. Zhang, Y. Xu, Y. Zhu, Q. Zang, Z. Wu, and Z. Wei, "Multiobjective task scheduling for energy-efficient cloud implementation of hyperspectral image classification," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 14, pp. 587–600, 2021.

[79] S. Bouzidi, "Parallel and distributed implementation on spark of a spectral–spatial classifier for hyperspectral images," *Journal of Applied Remote Sensing*, vol. 13, no. 3, pp. 1–18, 2019.

[80] A. Fernández, S. del Río, V. López, A. Bawakid, M. J. del Jesus, J. M. Benítez, and F. Herrera, "Big data with cloud computing: an insight on the computing environment, mapreduce, and programming frameworks," *WIREs Data Mining and Knowledge Discovery*, vol. 4, no. 5, pp. 380–409, 2014.

[81] ProjectPro, "Cloud computing vs. distributed computing." Available online: `https://www.projectpro.io/article/cloud-computing-vs-distributed-computing/94`, April 2022. (accessed on: 2022-04-10).

[82] C. Xu, X. Du, Z. Yan, and X. Fan, "Scienceearth: A big data platform for remote sensing data processing," *Remote Sensing*, vol. 12, no. 4, 2020.

[83] V. C. Gomes, G. R. Queiroz, and K. R. Ferreira, "An overview of platforms for big earth observation data management and analysis," *Remote Sensing*, vol. 12, no. 8, 2020.

[84] U. K.C., S. Garg, J. Hilton, J. Aryal, and N. Forbes-Smith, "Cloud computing in natural hazard modeling systems: Current research trends and future directions," *International Journal of Disaster Risk Reduction*, vol. 38, p. 101188, 2019.

[85] L. Wang, J. Yan, and Y. Ma, *Cloud Computing in Remote Sensing*. Chapman and Hall/CRC, 1 ed., July 2019.

[86] I. Zinno, S. Elefante, L. Mossucca, C. De Luca, M. Manunta, O. Terzo, R. Lanari, and F. Casu, "A first assessment of the p-sbas dinsar algorithm performances within a cloud computing environment," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 8, no. 10, pp. 4675–4686, 2015.

[87] M. N. Sadiku, S. M. Musa, and O. D. Momoh, "Cloud computing: Opportunities and challenges," *IEEE Potentials*, vol. 33, no. 1, pp. 34–36, 2014.

[88] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[89] The Apache Software Foundation, "Mapreduce tutorial." Available online: `https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html`, May 2022. (accessed on: 2022-06-04).

[90] Apache Hadoop, "Welcome to apache pig!." Available online: `https://pig.apache.org/`, Feb 2022. (accessed on: 2022-06-04).

[91] M. D. Assunção, R. N. Calheiros, S. Bianchi, M. A. Netto, and R. Buyya, "Big data computing and clouds: Trends and future directions," *J. Parallel Distrib. Comput.*, vol. 79-80, pp. 3–15, 2015. Special Issue on Scalable Systems for Big Data Management and Analytics.

[92] S. Roy, S. Gupta, and S. Omkar, "Case study on: Scalability of preprocessing procedure of remote sensing in hadoop," *Procedia Computer Science*, vol. 108, pp. 1672–1681, 2017. International Conference on Computational Science, ICCS 2017, 12-14 June 2017, Zurich, Switzerland.

[93] A. Gates and D. Dai, *Programming Pig: Dataflow Scripting with Hadoop*. O'Reilly Media, Inc., 2016.

[94] X. Tao, M. E. Paoletti, J. M. Haut, P. Ren, J. Plaza, and A. Plaza, "Endmember estimation with maximum distance analysis," *Remote Sensing*, vol. 13, no. 4, pp. 1–20, 2021.

[95] M. E. Winter, "N-findr: An algorithm for fast autonomous spectral end-member determination in hyperspectral data," in *Imaging Spectrometry V* (M. R. Descour and S. S. Shen, eds.), vol. 3753, pp. 266–275, International Society for Optics and Photonics, SPIE, 1999.

[96] V. A. Ayma Quirita, G. A. O. P. da Costa, and C. Beltrán, "A distributed n-findr cloud computing-based solution for endmembers extraction on large-scale hyperspectral remote sensing data," *Remote Sensing*, vol. 14, no. 9, 2022.

[97] V. H. Ayma, V. A. Ayma, and J. Gutierrez, "Dimensionality reduction via an orthogonal autoencoder approach for hyperspectral image classification," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLIII-B3-2020, pp. 357–362, 2020.

[98] V. Ayma, C. Beltrán, P. N. Happ, G. A. O. P. Costa, and R. Q. Feitosa, "Mapping glacier changes using clustering techniques on cloud computing infrastructure," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-2/W16, pp. 29–34, 2029.

[99] V. Ayma, C. Beltrán, P. Happ, G. Costa, and R. Feitosa, "Mapping ausangate glacier changes using clustering techniques on cloud computing infrastructure," in *Seventh International Conference on Remote Sensing and Geoinformation of the Environment (RSCy2019)*, vol. 11174 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, p. 111740L, 2019.

[100] Q. Du, N. Raksuntorn, N. H. Younan, and R. L. King, "Variants of n-findr algorithm for endmember extraction," in *Image and Signal Processing for Remote Sensing XIV*, vol. 7109, pp. 128–135, International Society for Optics and Photonics, SPIE, 2008.

[101] A. Remon, S. Sanchez, A. Paz, E. S. Quintana-Orti, and A. Plaza, "Real-time endmember extraction on multicore processors," *IEEE Geosci. Remote Sens. Lett.*, vol. 8, no. 5, pp. 924–928, 2011.

[102] L. I. Jiménez and A. Plaza, "Hypermix: An open-source tool for fast spectral unmixing on graphics processing units," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 9, pp. 1883–1887, 2015.

[103] X. Wu, B. Huang, A. Plaza, Y. Li, and C. Wu, "Real-time implementation of the pixel purity index algorithm for endmember identification on gpus," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 5, pp. 955–959, 2014.

[104] S. Sánchez, G. Martín, and A. Plaza, "Parallel implementation of the n-findr endmember extraction algorithm on commodity graphics processing units," in *2010 Int. Geosci. Remote Sens. Symp. (IGARSS)*, pp. 955–958, 2010.

[105] J. Setoain, M. Prieto, C. Tenllado, A. Plaza, and F. Tirado, "Parallel morphological endmember extraction using commodity graphics hardware," *IEEE Geosci. Remote Sens. Lett.*, vol. 4, no. 3, pp. 441–445, 2007.

[106] A. Paz and A. Plaza, "Clusters versus gpus for parallel target and anomaly detection in hyperspectral images," *EURASIP Journal on Advances in Signal Processing volume*, vol. 2010, no. 35, pp. 1–18, 2010.

[107] A. J. Plaza, J. Plaza, and A. Paz, "Parallel heterogeneous cbir system for efficient hyperspectral image retrieval using spectral mixture analysis," *Concurrency and Computation: Practice and Experience*, vol. 22, no. 9, pp. 1138–1159, 2010.

[108] A. Plaza, D. Valencia, J. Plaza, and P. Martinez, "Commodity cluster-based parallel processing of hyperspectral imagery," *J. Parallel Distrib. Comput.*, vol. 66, no. 3, pp. 345–358, 2006.

[109] A. Plaza, D. Valencia, J. Plaza, and C.-I. Chang, "Parallel implementation of endmember extraction algorithms from hyperspectral data," *IEEE Geosci. Remote Sens. Lett.*, vol. 3, no. 3, pp. 334–338, 2006.

[110] C. A. Lee, S. D. Gasster, A. Plaza, C.-I. Chang, and B. Huang, "Recent developments in high performance computing for remote sensing: A review," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, vol. 4, no. 3, pp. 508–527, 2011.

[111] Y. Chen, Z. Wu, Z. Wei, and Y. Li, "Pn-findr: A parallelized n-findr algorithm with spark," in *2016 International Conference on Advanced Cloud and Big Data (CBD)*, pp. 127–132, 2016.

[112] Z. Wu, J. Gu, Y. Li, F. Xiao, J. Sun, and Z. We, "Distributed parallel endmember extraction of hyperspectral data based on spark," *Scientific Programming*, pp. 1–9, 2016.

[113] J. Gu, Z. Wu, Y. Li, Y. Chen, Z. Wei, and W. Wang, "Parallel optimization of pixel purity index algorithm for hyperspectral unmixing based on spark," in *2015 Third International Conference on Advanced Cloud and Big Data*, pp. 159–166, 2015.

[114] P. Zheng, Z. Wu, W. Zhang, M. Li, J. Yang, Y. Zhang, and Z. Wei, "An unmixing-based content retrieval method for hyperspectral imagery repository on cloud computing platform," in *2018 Int. Geosci. Remote Sens. Symp. (IGARSS)*, pp. 3583–3586, 2018.

[115] The Apache Software Foundation, "Apache hadoop." Available online: `https://hadoop.apache.org/`, May 2022. (accessed on: 2022-06-04).

[116] A. Holmes, *Hadoop in practice*. Shelter Island, N.Y.: Manning Publications, 2015.

[117] T. White, *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 2015.

[118] A. Plaza and C.-I. Chang, "An improved n-findr algorithm in implementation," in *Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery XI* (S. S. Shen and P. E. Lewis, eds.), vol. 5806, pp. 298–306, International Society for Optics and Photonics, SPIE, 2005.

[119] J. M. Bioucas-Dias and J. M. P. Nascimento, "Hyperspectral subspace identification," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 8, pp. 2435–2445, 2008.

[120] Amazon Web Services, "Cloud computing services." Available online: `https://aws.amazon.com/`. (accessed on: 2022-04-10).

[121] Google, "Google cloud - cloud computing services." Available online: `https://cloud.google.com/`. (accessed on: 2022-04-10).

[122] Microsof, "Microsoft azure - cloud computing services." Available online: `https://azure.microsoft.com/en-us/`. (accessed on: 2022-04-10).

[123] IBM, "Ibm cloud platform." Available online: `https://cloud.ibm.com/`. (accessed on: 2022-04-10).

[124] Amazon Web Services, “Amazon ec2 m5 instances.” Available online: https://aws.amazon.com/ec2/instance-types/m5/, April 2022. (accessed on: 2022-04-10).

[125] D. Heinz and Chein-I-Chang, “Fully constrained least squares linear spectral mixture analysis method for material quantification in hyperspectral imagery,” *IEEE Trans. Geosci. Remote Sens.*, vol. 39, no. 3, pp. 529–545, 2001.

[126] N. Dimitri, “Pricing cloud iaas computing services,” *J Cloud Comp*, vol. 9, no. 14, 2020.