
Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
 Volume 25, Issues 6, pp. 5745–5759, DOI 10.12694/scpe.v25i6.4999

DATA CUBES AND CLOUD-NATIVE ENVIRONMENTS FOR EARTH OBSERVATION: AN OVERVIEW*

ALEXANDRU MUNTEANU†

Abstract. Reliable access to analysis-ready Earth observation data and infrastructures for processing them has been a challenge with the increasing volumes and variety of data being generated daily through various Earth observation programmes. Recently, concepts centered around building cloud-native infrastructures that provide access to Earth observation data in efficient manners such as data cubes which facilitate rapid querying, filtering and retrieval have been garnering popularity. Moreover, efficient means of processing such vast volumes of data stored in data cubes through cloud computing frameworks such as Kubernetes are becoming more popular. This paper investigates the current state-of-the-art techniques, methods and technologies used in cloud-native environments with a particular focus on the data cube initiative and "bring the user to the data" paradigm, highlighting the usefulness of such approaches and their current limitations.

Key words: Earth observation, data cubes, cloud-native, scalable computing

1. Introduction. With the rapid growth of Earth Observation (EO) data generated through programmes constantly deploying satellites to monitor the Earth's physical characteristics, efficient data management and processing strategies are needed. Public sector initiatives such as the European Space Agency (ESA) Copernicus programme and the National Aeronautics and Space Administration (NASA) Landsat or through private EO companies such as Planet Labs, Capella Space and many others are contributing to an unprecedented volume, variety and velocity of data regarding the physical characteristics of Earth daily. EO data provides valuable information, helping to develop strategies for a multitude of areas, such as climate change, disaster management, agricultural strategies, urban planning, and forest sustainability.

One of the principal challenges that arise when dealing with EO data comes from its complexity and heterogeneity. EO data comes in a variety of forms depending on the instrument used in the data acquisition phase and the processing techniques that are applied. These instruments range from optical, multi-spectral, hyper-spectral, RADAR, LiDAR and thematic instruments designed to capture information regarding the atmospheric composition, ocean and land colour and many others. In addition, it is worth mentioning that each data source provides data at different spatial resolutions and is disseminated through various formats (e.g. NetCDF [65], GeoTIFF [48], GeoParquet [67], Zarr [57] and others). Specific data processing workflows are employed based on this information.

With the advent of large, scalable infrastructures, especially Cloud Computing, High-Performance Computing (HPC) and distributed computing architectures, efficient processing of large volumes of EO data has become promising [83]. Frameworks for distributed computing such as Apache Spark [86] and Apache Hadoop [7] have facilitated processing and analyzing large datasets across clusters. Hadoop enables distributed storage (through HDFS - Hadoop Distributed File System) providing fault tolerance and high availability through data replication. By default, Hadoop uses the MapReduce paradigm, designed for batch processing the data stored in HDFS. Additionally, tasks are replicated across the cluster ensuring fault tolerance. Unlike Hadoop, Apache Spark uses in-memory data storage, which gives it an advantage in some use cases. In addition to batch processing, Spark supports real-time data streaming.

Kubernetes [12] is a cloud-native orchestrator for containerized applications in cloud environments, capable of deploying, scaling and managing containerized applications. The main advantages Kubernetes offers are

*Funding: This work was funded by the Romanian Ministry of Research, Innovation and Digitalization under contract no. PN-IV-P6-6.3-SOL-2024-2-0248, acronym ROCS.

†West University of Timișoara, Department of Computer Science (alexandru.munteanu@e-uvt.ro).

on-demand automatic scaling of deployments, rescheduling faulty containers, load balancing across containers or services, and efficient resource management.

A relatively recent initiative in managing large volumes of EO data consists of the development of Earth observation data cubes [38]. Earth observation data cubes are based on the data cube technology [10] where data is represented as multi-dimensional arrays that facilitate the process of querying, analysis and visualization of spatio-temporal data. In a typical data cube, metadata of the ingested products is kept within a DBMS, facilitating querying and filtering of the ingested products. In Earth observation data cubes, data is organised within multiple dimensions (e.g. latitude, longitude, time, spectral band). Recently, progress in standardising earth observation data cubes has driven current implementations to offer data that users can directly work with as part of what is known as Analysis-Ready Data (ARD) [43]. ARD proposes several preprocessing steps to be undertaken to ensure the quality of data delivered, thus creating data cubes that contain directly usable data.

Ongoing efforts through projects such as EOEPCA+[1] aim to standardize and design scalable architectures for supporting EO data processing. Other projects, such as Pangeo, PEPS, CODE-DE, EODC, Microsoft Planetary Computer, and Google Earth Engine, have deployed large-scale data dissemination and processing platforms which are hosted in scalable environments.

Copernicus Data Space Ecosystem (CDSE) is the most recent answer towards data cube approaches from the ESA. Data previously disseminated with the help of the now defunct Data Hub Software (DHuS) through ESA's ground segment and national replicas (also known as Collaborative Ground Segments - CollGS) are provided through CDSE. CDSE currently offers catalogue-based Application Programming Interfaces (API) such as STAC, OpenSearch, and OData, as well as non-catalogue APIs like OpenEO and OGC-compliant APIs. Data processing through On-Demand Processing (ODP) is also offered as part of CDSE through serverless functions.

Multi-mission algorithm and analysis platforms (MAAP) [6] is a joint ESA-NASA initiative designed to facilitate the analysis and processing of EO and in-situ data [5]. MAAP's implementation leverages open-source technologies and frameworks for developing a cloud-native approach to processing large-scale EO data. The principal reasoning behind MAAP is to "bring the user to the data" to reduce the significative overheads associated with data retrieval. Biomass harmonization and SAR data analysis are discussed by [29].

Integrating EO data cubes with scalable computing infrastructures, such as cloud platforms and HPC systems, has enhanced the ability to process and analyze large EO datasets. Architectures such as EOEPCA+ and cloud platforms such as the aforementioned Pangeo, CODE-DE, EODC, and CDSE all commonly offer user workspaces in cloud-based environments that are closer to the data to facilitate the scalable processing of data stored in their datacubes. Development Seed and Element84 employ cloud platforms like Amazon Web Services (AWS) to store and process large quantities of EO data.

In this article, we provide an overview of the state-of-the-art concerning the utilization of scalable infrastructures, architectures, technologies and practices for processing vast volumes of EO data, with a particular focus on approaches centred around using client-side Earth observation data cubes. We offer some insights regarding cloud-optimized data formats and the benefits of using them in cloud-native environments. We provide details about 8 different platforms that can be used for exploiting the potential offered through EO data cubes and information regarding the software environment or architectures those platforms use.

The paper is further organized in the following manner: Section 2 describes the current state-of-the-art in processing large volumes of Earth observation data, discussing modern HPC and cloud computing technologies employed by EO platforms. Platform architectures and undergoing standardization efforts are also taken into account. Furthermore, the various EO data cube developments are addressed in this section. Section 3 discusses the data cubes, platform standards and their current limitations. Finally, in Section 4, we draw our conclusions from this overview on the state-of-the-art of cloud-native environments for processing large volumes of EO data.

2. State of the Art. In a more generic term, the scientific community has discussed the use of scalable computing platforms for processing large volumes of data, particularly processing EO data stored in repositories following data cube approaches. In [9], the authors describe the use of current standards such as Spatio-Temporal Asset Catalog [71] and Open Data Cube (ODC) [38], as well as the use of distributed processing

[1] eoepca.org

methods such as Dask [15], Hadoop [7] or Apache Spark [86] for processing the large volumes of existing EO data. Highlighted by [9], the use of distributed computing can solve the scalability limitations of ODC raised by [82, 26]. Cloud-native data repositories for storing scientific data is a topic discussed by [3], highlighting the benefits of data-proximate computing and the use of cloud-native approaches to efficiently process large volumes of EO data.

The use of cloud-native approaches for analysing large volumes of Earth observation data, particularly with the data cube paradigm, has been a relatively recent development which has garnered popularity within the community, forming a solid ecosystem of standards, frameworks, platforms and software libraries [76].

A cloud-native approach towards defining processing pipelines for EO data cubes is described by [80] using the MapReduce [31] paradigm for processing Sentinel-2, 10m resolution products. Experimental results provided in [80] for performing land cover mapping at a continental scale using machine learning approaches based on Support Vector Machines (SVM) [32] and the U-Net [66] topology while using ESA WorldCover as the ground truth masks. The experiments were carried out within three different environments which facilitate both access to EO data cubes and computing infrastructures, namely Google Earth Engine (GEE) [28], Microsoft Planetary Computer [55] and the Science Earth Platform [81].

Two main limitations of the approach described by [80] are presented, namely that the implementation is highly complex, and users are required to manually define the dependencies on which the data cubes are built. Secondly, the range of algorithms that can be applied to the generated data cube is limited due to how the data cube is partitioned. Algorithms such as Principal Component Analysis (PCA) cannot be easily implemented using this approach [80].

2.1. Earth Observation Data Cubes. Various methods for creating data cubes with EO data have been broadly discussed in the literature [24, 25, 40, 74]. In [40], the authors discuss "achieving the full vision of Earth observation data cubes", where the prerequisites and methodology for building EO data cubes are outlined, most notably the data preprocessing steps for building ARD [43] according to the Committee on Earth Observation Satellites (CEOS)[2] CARD4L guidelines [1].

According to the CEOS CARD4L guidelines [1], a series of processing steps need to be performed on the data before dissemination. Namely, radiometric and geometric preprocessing, tiling, compression, choosing a well-suited data format, generation of multiple overview layers, and optimizing the data for temporal access [40].

Optimizing data storage using compression and choosing data formats and structures that optimize access to the data are also discussed by [40]. The addition of processing workflows, user workspaces and the ability to disseminate the data and value-added products is also highlighted by [40]. The benefits of Combining analytic interfaces with EO data cubes for facilitating the execution of processing workflows are discussed by [49] covering three use cases: analyzing the statistics of biosphere-atmosphere interactions, the dynamics of intrinsic dimensions of ecosystems and model parameter estimation.

The benefits of local or national level EO data cubes are highlighted by [75]. Thematic data cubes such as CBERS [64] designed for mapping biomes in Brazil or mapping agriculture [13] require smaller infrastructures to manage, reducing the load of more general purpose EO data cubes at the cost of not having all the data conveniently in the same platform, difference in technologies and choice of standards. This spans the need for federating access to various data cubes or platforms, fitting into the vision of the EOEPCA+ architecture.

Earth observation data cubes have been employed for solving various tasks such as mapping surface water over a temporal span of 25 years [59] using the AGDC, developing machine learning based time series analysis packages for the R language [70], rapid high-resolution detection of environmental changes at continental levels [44]. These use cases highlight the relevancy and importance of further developing such standardized, cloud-native approaches for processing large-scale EO data.

2.1.1. Cloud-Native Geospatial Data Formats. One of the central points of building cloud-native EO data cubes is the conversion of data from their various initial formats to cloud-friendly formats that facilitate random access and partial file reads over various protocols such as the HyperText Transfer Protocol (HTTP).

Particularly, the development of the Cloud Optimized GeoTIFF (COG)[3] format for raster data which

[2]https://ceos.org
[3]https://cogeo.org

Table 2.1: Cloud-native formats for storing vector data.

Format	Base format	Structure and optimization
COPKG	GeoPackage	SQLite, HTTP range requests
GeoParquet	Parquet	Columnar data layout, spatial indexing
FlatGeoBuf	Flatbuffers	Packed Hilbert R-tree [37], HTTP range requests
Geojson-T	GeoJSON	Tiled GeoJSON, partial retrieval

organises pixels into tiles which are indexed (using an offset table) for rapid access, with multiple generated pyramids acting as overview layers. Each tile within a COG file can be individually compressed, with popular choices being LZW, Deflate or JPEG compression algorithms. Performing partial file reads is possible for COG files via HTTP GET range requests[4] that correlate with the random access indexed tiles provide. Cloud-Optimized GeoTIFF files can be conveniently created with the help of Rasterio [23], a Python library that handles raster geospatial data. More precisely, with the use of the `rio-cogeo`[5] plugin.

A similar format for storing 3D point cloud data is the Cloud Optimized Point Cloud (COPC) format[6] which is based on the LIDAR Aerial Survey (LAZ) format. COPC files share a similar partial file, random access vision as COG which is implemented using an Octree [68] data structure. Similar to COG, COPC files also have overview layers computed also known as Levels of Detail (LOD). In terms of compression, LZW is typically used with COPC data. HTTP range requests can be used to access nodes from the Octree representation, allowing for partial reads.

In terms of cloud-native formats for storing vector geospatial data, due to the availability of multiple formats in existence (Apache Parquet[7], FlatGeoBuf, GeoJSON, ESRI Shapefile, Apache Arrow) paired with a lack of consensus in the community have led to the development of multiple suitable formats. Most notably, formats such as Cloud Optimized GeoPackage (COPKG), GeoParquet, Geojson-T (Tiled GeoJSON) and Mapbox Vector Tiles (MVT). Through PMTiles[8], support for HTTP range requests and generation of COG-like pyramids is aimed to be brought to vector data formats as well [78]. An approach for cloud-optimized tile archive formats deployed in the cloud is presented in [78]. Similar efforts towards raster encodings for web-native for time series data designed for large environmental EO data in use for streaming in web platforms are discussed by [34]. Table 2.1 contains popular cloud-native vector formats, the format they are based on, and their indexing method.

Among the formats shown in Table 2.1, GeoParquet has advantages over the others in terms of compression, querying speed and throughput [67, 53, 79] and is used in platforms such as Microsoft Planetary Computer [55].

2.1.2. Current Operational EO Data Cubes. With the development of the Australian Geosciences Data Cube (AGDC) in 2017 [45], the Open Data Cube (ODC) initiative was spanned [38]. An overview of the deployed ODC instances[9] in 2018 [38] discusses that at the time, four national instances were already operational: Switzerland [24], Columbia [8], Taiwan [14] and Australia [45] with 11 others being in development. Later developed instances such as the Austrian Semantic Data Cube [75], the CBERS data cube for mapping biomes in Brazil [64], the Romanian Data Cube [62] rely on the use of the Spatio-Temporal Asset Catalog specification[10] and cloud-optimized data storage formats which forms the new direction dissemination of Earth observation data is heading towards. Recent versions of ODC have also adopted the STAC specification[11] and provide access to data through compliant API's. Table 2.2 shows some of the currently deployed data cubes, their spatial coverage and URL's where further details and access methods can be consulted.

[4] https://tools.ietf.org/html/rfc7233
[5] https://cogeotiff.github.io/rio-cogeo/
[6] https://copc.io
[7] https://parquet.apache.org
[8] https://cloudnativegeo.org/blog/2023/10/where-is-cog-for-vector/
[9] https://opendatacube.readthedocs.io/
[10] https://stacindex.org/catalogs
[11] https://www.opendatacube.org/copy-of-get-started

Table 2.2: Current deployments of Earth Observation data cubes.

Name	Coverage	URL
MPC	Global	https://planetarycomputer.microsoft.com/
GEE	Global	https://earthengine.google.com
GEO	Global	https://www.earthobservations.org
AGDC	Australia	https://www.ga.gov.au/dea
SDC	Switzerland	https://www.swissdatacube.org/
ACUBE	Austria	https://acube.eodc.eu
eocube.ro	Romania	https://eocube.ro/
CBERS	Brazil	https://brazil-data-cube.github.io
TASA	Taiwan	https://www.tasa.org.tw
Digital Earth Africa	Africa	https://www.digitalearthafrica.org/
Digital Earth Pacific	Pacific Islands	https://www.digitalearthpacific.org/
INEGI	Mexico	http://en.www.inegi.org.mx
Armenian	Armenia	http://datacube.sci.am
SIBELIUs	Mongolia, Kyrgyzstan	https://eosphere.co.uk
SERVIR	Mekong Region	https://servir.adpc.net

2.2. HPC for Processing Large Volumes of EO Data. The adoption of High-Performance Computing (HPC) has been discussed largely by [46] where authors discuss traditional general-purpose HPC frameworks such as Apache Spark [85], Hadoop [7], OpenMPI [21] and HTCondor [77] and their respective use in conjunction with Earth observation data. Particularly, the use of HPC and cloud computing resources for processing EO data organized in data cubes is addressed by [9] through the use of Dask [15] clusters orchestrated by Kubernetes [12]. The authors of [9] present an architecture leveraging those technologies to exploit EO data cubes that utilise the Spatio-Temporal Asset Catalog (STAC) [71] specification.

A software solution tailored especially for processing large quantities of geospatial data based on Spark is Apache Sedona (formerly known as GeoSpark) [85]. Sedona stores classical georeferenced vector data types such as points, lines, linestrings and polygons in custom Spatial Resilient Distributed Datasets (SRDD). Furthermore, Sedona utilises spatial indexing data structures such as R-trees and Quad-trees, enabling efficient queries. Queries based on relationships and geospatial functions are also implemented in Apache Sedona. Sedona is fully integrated with the Apache Spark ecosystem, allowing the use of Spark SQL, Spark Core and Spark DataFrames.

GeoTrellis[12] is a geospatial processing engine designed for execution in HPC environments. Developed on top of Apache Spark, GeoTrellis can be deployed in cluster and grid environments, allowing it to scale to fit various processing requirements. GeoTrellis supports processing both vector and raster data, it provides raster operations (map algebra), spatial operations and utilities that facilitate the creation of web services for disseminating the processed products [42]. Some limitations of GeoTrellis include two resampling techniques (Nearest Neighbor and Bilinear sampling) that affect the runtime of the overall process, as well as having no control over processing steps and job scheduling, therefore relying only on Spark's scheduling [42]. Raster processing using GeoTrellis is discussed in [41], where a cloud architecture is proposed, leveraging the use of Docker [52] to distribute the workload in a cluster.

Google BigQuery [11] is a serverless, scalable data warehouse product from the Google Cloud. With on-demand scaling, and serverless architecture there is no need of infrastructure management. BigQuery utilises a columnar format for data storage that is separate from the compute capabilities, data retrieval is done through an SQL-like language. In the context of a comprehensive comparative study for large geospatial data storage methods [16], both the benefits and disadvantages of BigQuery we're detailed. Integration with other Google services, reliability and serverless architecture, ease of use and standard SQL querying capabilities are mentioned as the strong points of BigQuery [16]. The financial model of pay-as-you-go requires cost monitoring by the users, varying ingestion rates and highly complex geospatial data might not benefit from the NoSQL

[12]https://geotrellis.io

architecture BigQuery employs constitutes the drawbacks [16].

Dask [15] is an open-source library for parallel computing that facilitates scaling Python applications for various tasks such as data processing, machine learning, and distributed computing. It is optimised to work well on large datasets and is a scalable technology, with the possibility of deploying Dask as a cluster. Dask can perform distributed computations with nd-arrays, which perfectly aligns with processing EO data. Dask-GeoPandas[13] adds support for partitioning geospatial data into spatially distributed chunks and facilitates the parallelization of spatial operations. Dask utilises a dynamic task scheduler to execute computations, making it well-suited for complex workflows efficiently. A Dask cluster is part of the Pangeo [2] project being available interactively within the user workspaces via Jupyter Notebooks [39].

Simple Linux Utility for Resource Management (SLURM) [84] is a workload manager and job scheduling system that efficiently manages resource allocation within clusters. SLURM allows for job scheduling using fair scheduling algorithms. SLURM includes job a dependency system, enabling for scheduling complex workflows. Task scheduling for three separate use cases for processing large volumes of EO data on the EODC platform was performed through SLURM [17]. The Pangeo project can also be configured to use the SLURM scheduler [2].

2.3. EO Data Exploitation Platforms. In the context of providing both EO data and access to nearby computing resources for processing data, several projects have been recently developed [76], most notably EOEPCA+, Pangeo [2], CODE-DE [72], PEPS [22], EODC[14], CDSE [56], Microsoft Planetary Computer [55], Google Earth Engine [28], and Amazon Web Services[15]. All these projects leverage the use of cloud computing and, in some cases, HPC for processing large-scale Earth observation data, which is organized within a data cube approach. This section briefly details the platform's methodologies, architectures, standards and software frameworks.

The goal of the Earth Observation Exploitation Platform Common Architecture (EOEPCA+)[16] project is to bring standardisation and federation by designing a cloud-native architecture in line with best practices in software engineering, aiming to facilitate the way EO data is processed. EOEPCA+ is currently developing an open-source implementation of the architecture's components. This architecture is divided into three layers, as shown in Figure 2.1.

The **platform layer** is comprised of microservices designed for data discovery and ingestion, running various processing workflows to generate added-value products. Within this layer, workspaces for users are also running, offering persistance, access to EO data cubes, visualization capabilities and code execution for users to process the available data further. This layer contains processing engines, which facilitate the execution of various user-defined workflows such as openEO Process Graphs[17] and OGC Application Packages[18].

The goal of the **federation layer** is coordinating access towards multiple platforms. A federated orchestrator can direct processing workflows to the appropriate platform (i.e., one that meets the workflow requests, is currently available in terms of resources, etc.). Resource discovery integrates cross-platform data catalogues, facilitating data querying capabilities amongst the platforms. Identity and access management is also coordinated at this layer, redirecting users towards their use spaces The Storage Controller at this layer, besides managing the platform's storage, allows for external storage services to be integrated into the user workspace, achieving data federation.

Lastly, the **application layer** contains interactive web-based tools for users to publish web dashboards and applications to disseminate the results of processing the data provided through the platform. The application layer facilitates the definition of processing workflows, executed within testing environments on the platform.

Projects such as Pangeo [2] have employed Kubernetes and have designed cloud-native approaches for processing large volumes of EO data using Dask [15] and Xarray [33]. The use of Zarr and Xarray instead of traditional NetCDF/HDF for storing Earth Observation data for facilitating it's use in cloud-native environments is discussed by [3] and [4]. Pangeo can be deployed in traditional HPC infrastructures [63] such as NASA

[13]https://dask-geopandas.readthedocs.io
[14]https://eodc.eu
[15]https://aws.amazon.com
[16]https://eoepca.org
[17]https://api.openeo.org/#section/Processes/Process-Graphs
[18]https://docs.ogc.org/bp/20-089r1.html
[19]https://eoepca.readthedocs.io/

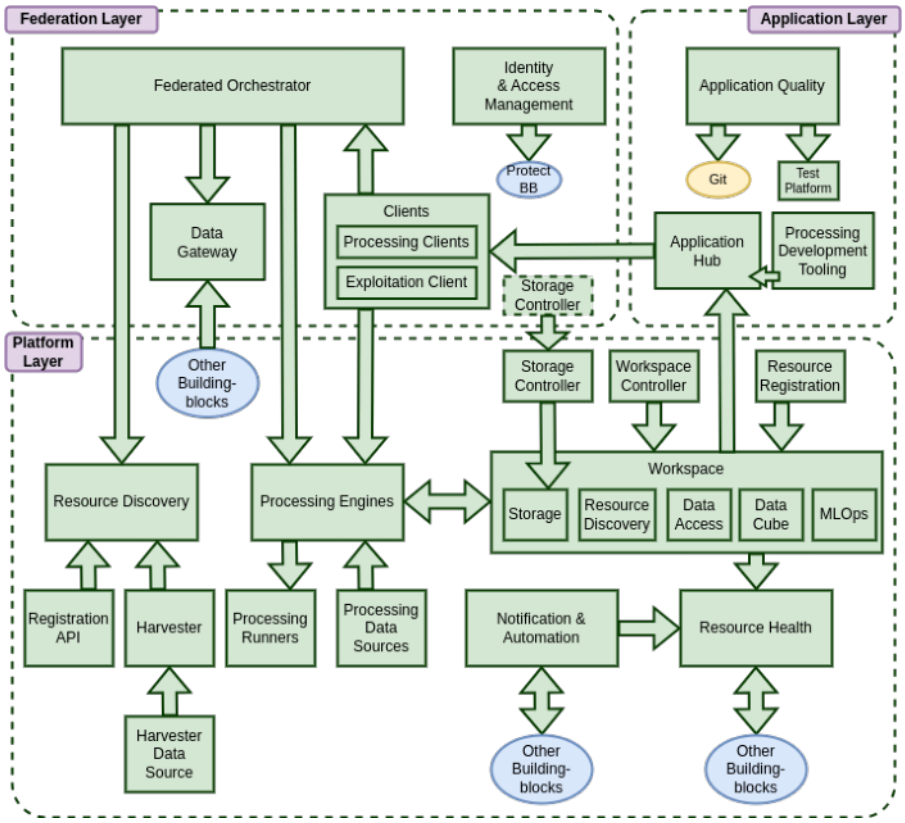

Fig. 2.1: EOEPCA+ High-level architecture[19].

Pleiades[20], Cheyenne from NCAR[21], Google Cloud Platform or Amazon Web Services. Within the Pangeo project, educational interactive resources can be accessed through provided workspaces running in Jupyter notebook environments. Use of the Pangeo project at the Centre National d'études Spatiales (CNES)[22] is described by [18], showcasing its usefulness and ease of use for processing data using HPC resources with Dask. One of the use cases CNES employs Pangeo for, namely numeric computations for analysing surface ocean currents on a large scale. Unfortunately performance assesments of Pangeo for this task are not provided by [18].

Figure 2.2 illustrates the Pangeo architecture. As aforementioned, the use of cloud object storage for serving chunked data with the Zarr format, coupled with querying capabilities provided through Xarray. This data is accessed through microservices running in a compute cluster orchestrated by Kubernetes, providing users with interactive notebooks and access to a Dask cluster that facilitates parallel processing.

Copernicus Data and Exploitation Platform - Deutschland (CODE-DE) [72, 73] is a platform built for disseminating EO data for the German authorities as a collaborative ground segment, developed concerning various user requirements elaborated by the German Aerospace Center (DLR). The CODE-DE platform was designed to suit multiple needs, such as project management, product assurance, systems engineering, data ingestion and archiving, querying and retrieval, processing environments, storage and dissemination of value-added products derived from raw Sentinel data [73].

The CODE-DE platform enables registered users to access various data processors and processing workflows,

[20]https://www.nas.nasa.gov/hecc/resources/pleiades.html
[21]https://www.cisl.ucar.edu/ncar-wyoming-supercomputing-center
[22]https://cnes.fr/en

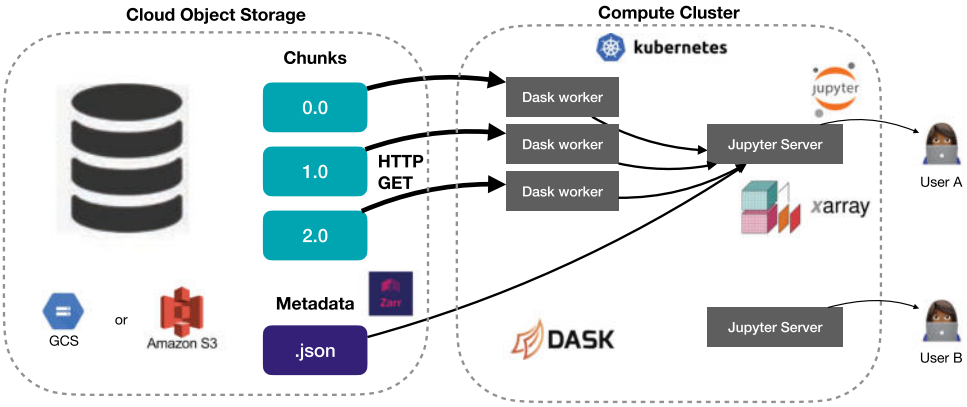

Fig. 2.2: The cloud-native architecture of Pangeo [3].

enabling them to independently process Earth observation data using selected methodologies and subsequently disseminate the resulting value-added products. The processing environment available in CODE-DE supports algorithm selection and spatial queries for specific EO datasets while also allowing users to monitor the current status of their processing tasks. Among the available methods are tools from the Sentinel toolbox, such as Sen2Cor [50], employed for the atmospheric correction of Sentinel-2 Level 1C products [72].

Users can interact with the platform through a web interface or via various APIs (OpenSearch or OGC-compliant services[23] like WMS, WFS, WCS). The data catalogue integrates with the various API's and is exposed to the user via a web application. Metadata for data collections is interactively generated using ISO standards and is accessible via OGC-compliant Catalogue Service for the Web (CSW). For products, metadata is automatically generated following OGC EOP[24] standards. CODE-DE services are modular and adhere to the INSPIRE conform discovery, visualization and download standards. Data processing workflows in the CODE-DE platform are executed through Calvalus [20] or Apache Hadoop [7]. They can be described and triggered either through a web application or through an OGC Web Processing Service (WPS) API [72]. The CODE-DE architecture is illustrated in Figure 2.3.

Plateforme d'exploitation des produits Sentinel (PEPS) [22] is CNES's solution towards providing access to Sentinel data as part of the Copernicus programme, PEPS is a member of the ESA collaborative ground segment. PEPS offers a web interface that enables users to query, filter, choose data preprocessing tasks and retrieve raw or value-added Sentinel-1, Sentinel-2 and Sentinel-3 products. Querying and filtering products in the PEPS platform is achieved through RESTO[25] catalogues [22].

PEPS offers several online data processing tools aimed at creating value-added products reducing download sizes (i.e. downloading results, not entire datasets) and performing preprocessing tasks, allowing users to access ready-to-analyze data. A couple of data processing capabilities are included in PEPS, such as computing Normalized Difference Vegetation Indices (NDVI), polarization extraction, atmospheric corrections for Sentinel-2 data using the MACCS-ATCOR Joint Algorithm (MAJA) [47], water masks generation, extraction of metadata and ortho-rectification.

PEPS services are executed on an HPC infrastructure in a containerized environment facilitated through Docker [52] containers [22]. An implementation of OGC Web Processing Service (WPS) [58] facilitates the definition of processing workflows which are scheduled for execution using the PROACTIVE Meta Scheduler[26] in conjunction with the Portable Batch System (PBS) [36]. The PEPS platform facilitates access to large-scale Earth observation datasets, which can integrate with external platforms or processing pipelines. The PEPS

[23]https://www.ogc.org/standards
[24]https://docs.ogc.org/is/10-157r4/10-157r4.html
[25]https://github.com/jjrom/resto
[26]https://proactive.activeeon.com

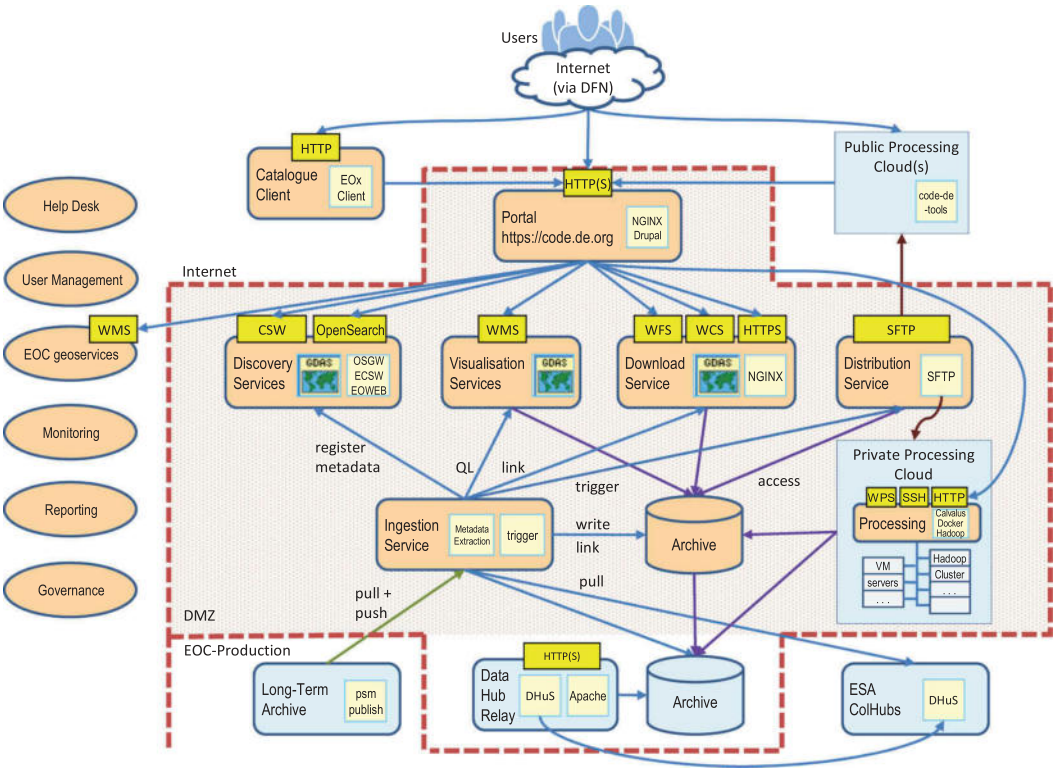

Fig. 2.3: The CODE-DE architecture [72, 73].

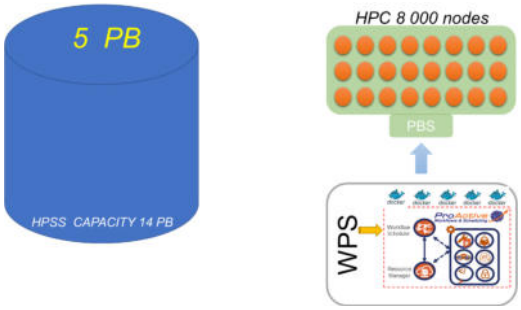

Fig. 2.4: The PEPS Architecture [22].

platform architecture is illustrated in Figure 2.4.

The Copernicus Data Space Ecosystem (CDSE) [56] is ESA's principal platform for disseminating data acquired through the Copernicus programme. Federated access, user identity, data access and visualization are all discussed from multiple perspectives (data providers, remote sensing experts, application developers, platform integrators and governance) by [60]. Through Jupyter Notebooks, interactive user workspaces are available within the CDSE. The workspaces are integrated [60] with the OpenEO framework [35] which promotes federation and makes use of distributed computing environments and enables the definition of processing workflows for big EO datasets. Multiple Data and Information Access Services (DIAS) are linked with the CDSE, allowing access to cloud resources that facilitate access to the EO data and offer VPS-based computing capabilities. Sentinel Hub, a processing service for EO data designed for on-the-fly computations, is also

integrated with the user workspaces [60].

An overview [60] of the API's provided for accessing CDSE. The Open Data Protocol (OData), OpenSearch and STAC are all offered. One important feature incorporated in CDSE, especially of interest to CollGS, is the notification API, which enables the registration of webhooks that are called when new products are added to collections of interest.

The Earth Observation Data Centre (EODC) provides services for accessing and processing EO data while providing compute resources based on virtualization. The use of the EODC platform for processing large volumes of EO data in a general sense is described by [17]. EODC was also utilised for retrieving geophysical parameters from Sentinel-1 Synthetic Aperture Radar (SAR) data by [61]. Although offering cloud computing resources, EODC follows a Virtual Private Server (VPS) approach for renting virtualized environments without the possibility of on-demand scaling.

Google Earth Engine (GEE) [28] is another cloud-based platform that facilitates EO data analysis, visualization and processing. By leveraging the Google Cloud infrastructure, GEE enables for processing of large datasets. The data catalogue currently contains Landsat, Sentinel, MODIS, climate data, land use and land cover (LULC), air quality, and other georeferenced datasets. The STAC specification was also adopted for the data catalogue. Interactive user workspaces are provided within GEE, allowing for JavaScript code execution and visualization. Furthermore, due to Google's rich ecosystem, interactive access through Google Colab offers the possibility of interacting with the Earth Engine as well. Programmatic access to GEE is possible via Python and JavaScript API's.

Microsoft Planetary Computer (MPC) [55] offers similar capabilities as GEE with a rich data catalogue focusing on biodiversity, environmental and ecological data. This data catalogue is also exposed using the STAC specification, leverages the Zarr [57] format, and serves vector data under the GeoParquet [67] format. The Planetary Computer provides users with workspaces through interactive Jupyter Notebooks. Dask is also provided within the workspace to distribute large processing workloads. Users can leverage Microsoft Azure's cloud computing power for large-scale environmental analysis, which is particularly beneficial for handling large datasets like global satellite imagery and climate models. Integration with Azure AI allows the use of pre-trained Machine Learning models with the data found in the Planetary Computer catalogue. The Planetary Computer is also integrated with Azure Blob Storage, allowing for the easy storage of processing results.

The Amazon Web Services (AWS) cloud infrastructure is a popular choice for private sector companies that process EO data, such as DevelopmentSeed and Element84. Like GEE and MPC, AWS benefits from a large ecosystem of technologies for storing and processing data in cloud environments. The use of AWS for improving land use and land cover mapping in Brazil is addressed by [19]. By leveraging serverless (AWS Lambda) functions, object storage (AWS Buckets), Tile Map Services and DevelopmentSeed's implementation of STAC catalogues[27], [19] have developed a platform for forest monitoring. A serverless land evaluation platform designed by [54] Amazon Elastic Compute Cloud (EC2) [69] was integrated with an OGC WPS compliant implementation [87] for EO data processing. Furthermore, NASA HPC workflows have been evaluated with EC2 [51] and compared to NASA's Pleiades infrastructure.

Table 2.3 illustrate the various cloud-native platforms for processing EO data, utilising a data cube approach for serving data.

3. Discussion. Current deployments of platforms that leverage the potential of cloud computing resources for processing large volumes of Earth observation data share some common traits. Undergoing standardisation efforts taken by initiatives such as EOEPCA+ aim to bring those platforms as interoperable and federalised as possible while following best practices from software engineering and geospatial data perspectives.

Platforms like Pangeo [2] leverage the Kubernetes [12] orchestrator for scalable deployment, efficient resource management, and on-demand scaling of distributed computing environments, facilitating efficient processing of Earth observation data. PEPS [22] employs a containerised approach using Docker [52] for managing the platform's components.

[27]https://sat-api.developmentseed.org/search/stac
[30]Implemented as modules in Pangeo and can be utilised depending on the available infrastructure.
[30]https://altair.com/pbs-professionalg
[30]Users can deploy their own frameworks on the VPS.

Table 2.3: EO Platforms.

Platform	Workflows	Workspaces	Distributed Processing
Pangeo	yes	JupyterHub	Dask, Slurm, Spark, YARN[28]
CODE-DE	yes	JupyterHub	Hadoop, Docker
PEPS	yes	N/A	CNES HPC (PBS)[29], ProActive
CDSE	yes	JupyterHub	Through OpenEO, SentinelHub
EODC	yes	N/A	yes[30]
MPC	yes	JupyterHub	Dask
GEE	yes	yes	GCP
AWS	yes	yes	EC2

Support for distributed computing frameworks such as Apache Spark [86], Apache Hadoop [7], Google BigQuery [11] or variants built for EO data such as Apache Sedona [85], GeoTrellis and more commonly seen in the platforms mentioned in Section 2, Dask [15]. Microsoft Planetary Computer [55] employ Dask for distributed computing workflows. CODE-DE makes use of Hadoop [7], while Pangeo's [2] versatile modules can integrate with Dask [15], SLURM [84] and Spark [85]. The PEPS platform utilises the ProActive scheduler to manage jobs executed on CNES's HPC cluster using PBS. Additionally, platforms integrated into larger cloud ecosystems, such as Google Earth Engine [28], Microsoft Planetary Computer [55], and Amazon Web Services, have developed in-house tools for big data processing workflows.

Workspaces in which users can explore data catalogues, define and submit processing workflows which are executed through schedulers such as SLURM [84], or use Dask [15]'s integrated job scheduling system. The majority of platforms described in this overview (Pangeo, CODE-DE, CDSE and Microsoft Planetary Computer) provide interactive workspaces through JupyterHub [39], which allows to write and execute Python code near the data. These workspaces typically include access to API's, libraries or SDK's for distributed or parallel processing frameworks. The joint ESA-NASA initiative of MAAP [5] aims to bring user workspaces close to the data by providing a cloud-based platform where users can access, analyse, and visualise big Earth observation datasets in a collaborative environment.

The choice of a standard, cloud-friendly data format such as Cloud-Optimized GeoTIFF (COG), Cloud-Optimized Point Cloud (COPC) and object storage makes partial file reads possible using HTTP range requests while also reducing the amount of data that needs to be downloaded to specific areas of interest of the users. Although multiple cloud-friendly formats for vector data have been designed, many Earth observation data cubes use GeoParquet [67] format due to its advantages [53, 79].

The Spatio-Temporal Asset Catalog [71] specification has been adopted by the majority of the platforms described in Section 2. Google Earth Engine [28], Microsoft Planetary Computer [55], Copernicus Data Space Ecosystem [56], CODE-DE [73] all expose data catalogues using the STAC specification. An issue with the STAC-compliant API offered through CDSE is incomplete product metadata. STAC extensions such as `eo, sat, sar, mgrs`[31] are not yet supported through this API.

The CDSE [56] implementation of federalisation for user access for data access, among other platforms like DIAS and processing workflows, ensures the availability of data and processing capabilities at all times.

4. Conclusions. In this paper, we have presented an overview of the current state of the art in Earth observation data cubes, focusing on cloud-native platforms designed for exploiting such resources. Several High-Performance Computing and cloud computing frameworks, job schedulers, and orchestrators, such as Apache Spark, Apache Hadoop, Dask, SLURM, Apache Sedona, Kubernetes and Docker, are briefly discussed, highlighting their importance in efficient processing and management of large volumes of Earth observation data.

Their implications in architectures for developing platforms that leverage the potential of EO data, such as EOEPCA+, Pangeo, PEPS, CODE-DE, Copernicus Ecosystem Data Space, EODC, Microsoft Planetary Computer, and Google Earth Engine, are paramount for facilitating the efficient processing of large volumes of

[31] https://stac-extensions.github.io

data that are being generated at unprecedented volumes. Earth observation data cubes particularities, cloud-friendly data formats, and currently deployed instances that serve collections amounting to petabytes of data daily were briefly discussed.

This overview has shown a strong shift towards leveraging cloud-native principles such as microservices, orchestration, containerisation, serverless computing and horizontal scaling in large Earth observation platforms. Additionally, the use of object storage for hosting products in cloud-optimized formats which facilitate the transfer of data and integrate well with specifications such as Spatio-Temporal Asset Catalog has become increasingly popular, with multiple platforms disseminating Earth Observation data in this manner.

Though COG and COPC are utilised "de facto" in EO data cubes, the lack of a consensus for vector data formats currently requires the use of different libraries and technologies capable of ingesting and processing multiple formats. However, GeoParquet [67] has garnered popularity among platforms such as and could become the most adopted format for vector data due to its data representation, ability to host large amounts of information, and ease of querying [53, 79].

This overview has shown that significant standardisation efforts, such as those undertaken through initiatives such as EOEPCA+, Open Data Cube, and Spatio-Temporal Asset Catalog, are essential for integrating various platforms and data sources. Federalisation efforts are paramount within such large ecosystems to ensure interoperability amongst platforms, seamless data dissemination, and collaboration across various institutions.

However, this overview has only paved the way for analysing these platforms' potential for processing large volumes of Earth observation data. Overviews on data access platforms such as [27] or data cube initiatives [38, 30], surveys on or individual insights regarding platforms [22, 72, 73, 2, 63] designed for processing big Earth observation data all contribute valuable information towards shaping the current state-of-the-art and the directions in which Earth observation platforms are headed. Inventoring software through collaborative initiatives such as OSS4GEO[32] aim to create a knowledge base for open source technologies developed for geospatial data exploitation. Comparative studies from technological standpoints, scoping reviews, and more in-depth studies should be considered and further developed to better understand the potential and limitations of Earth observation platforms.

REFERENCES

[1] *CEOS Analysis Ready Data for Land (CARD4L) Overview*.

[2] R. Abernathey, K. Paul, J. Hamman, M. Rocklin, C. Lepore, M. Tippett, N. Henderson, R. Seager, R. May, and D. Del Vento, *Pangeo nsf earthcube proposal*, (2017).

[3] R. P. Abernathey, T. Augspurger, A. Banihirwe, C. C. Blackmon-Luca, T. J. Crone, C. L. Gentemann, J. J. Hamman, N. Henderson, C. Lepore, T. A. McCaie, et al., *Cloud-native repositories for big scientific data*, Computing in Science & Engineering, 23 (2021).

[4] R. P. Abernathey, J. Hamman, and A. Miles, *Beyond netCDF: Cloud Native Climate Data with Zarr and XArray*, in AGU Fall Meeting Abstracts, vol. 2018, 2018, pp. IN33A–06.

[5] C. Albinet, A. S. Whitehurst, L. A. Jewell, K. Bugbee, H. Laur, K. J. Murphy, B. Frommknecht, K. Scipal, G. Costa, B. Jai, et al., *A joint esa-nasa multi-mission algorithm and analysis platform (maap) for biomass, nisar, and gedi*, Surveys in Geophysics, 40 (2019), pp. 1017–1027.

[6] C. Albinet, A. S. Whitehurst, H. Laur, K. J. Murphy, B. Frommknecht, K. Scipal, A. E. Mitchell, B. Jai, and R. Ramachandran, *Esa-nasa multi-mission analysis platform for improving global aboveground terrestrial carbon dynamics*, in IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2018, pp. 5282–5284.

[7] Apache Software Foundation, *Hadoop*.

[8] C. Ariza-Porras, G. Bravo, M. Villamizar, A. Moreno, H. Castro, G. Galindo, E. Cabera, S. Valbuena, and P. Lozano, *Cdcol: A geoscience data cube that meets colombian needs*, in Advances in Computing: 12th Colombian Conference, CCC 2017, Cali, Colombia, September 19-22, 2017, Proceedings 12, Springer, 2017, pp. 87–99.

[9] H. Astsatryan, A. Lalayan, and G. Giuliani, *Scalable data processing platform for earth observation data repositories*, Scalable Computing: Practice and Experience, 24 (2023), pp. 35–44.

[10] P. Baumann, P. Furtado, R. Ritsch, and N. Widmann, *The RasDaMan approach to multidimensional database management*, in Proceedings of the 1997 ACM Symposium on Applied Computing - SAC '97, ACM Press, 1997, pp. 166–173.

[11] E. Bisong and E. Bisong, *Google bigquery*, Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners, (2019), pp. 485–517.

[12] E. A. Brewer, *Kubernetes and the path to cloud native*, in Proceedings of the sixth ACM symposium on cloud computing, 2015, pp. 167–167.

[32] https://project.oss4geo.org

[13] M. E. D. Chaves, A. R. Soares, I. D. Sanches, and J. G. Fronza, *CBERS data cubes for land use and land cover mapping in the Brazilian Cerrado agricultural belt*, International Journal of Remote Sensing, 42 (2021), pp. 8398–8432.
[14] M.-C. Cheng, C.-R. Chiou, B. Chen, C. Liu, H.-C. Lin, I.-L. Shih, C.-H. Chung, H.-Y. Lin, and C.-Y. Chou, *Open data cube (odc) in taiwan: The initiative and protocol development*, in IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2019, pp. 5654–5657.
[15] Dask Development Team, *Dask: Library for dynamic task scheduling*, 2016.
[16] V. S. V. Deepika, K. B. Sri, V. Katyayani, G. Sahitya, and V. Rachapudi, *A comprehensive study of geospatial data storage mechanisms*, in 2024 International Conference on Expert Clouds and Applications (ICOECA), IEEE, 2024, pp. 87–95.
[17] S. Elefante, V. Naeimi, S. Cao, I. Ali, T. Le, W. Wagner, and C. Briese, *Big data processing using the eodc platform*, in Proceedings of the 2017 conference on Big Data from Space (BiDS´ 17), Publications Office of the European Union, 2017, pp. 9–12.
[18] G. Eynard-Bontemps, R. Abernathey, J. Hamman, A. Ponte, and W. Rath, *The pangeo big data ecosystem and its use at cnes*, in Big Data from Space (BiDS'19).... Turning Data into insights... 19-21 february 2019, Munich, Germany, 2019.
[19] K. R. Ferreira, G. R. Queiroz, G. Camara, R. C. M. Souza, L. Vinhas, R. F. B. Marujo, R. E. O. Simoes, C. A. F. Noronha, R. W. Costa, J. S. Arcanjo, V. C. F. Gomes, and M. C. Zaglia, *Using Remote Sensing Images and Cloud Services on Aws to Improve Land Use and Cover Monitoring*, in 2020 IEEE Latin American GRSS & ISPRS Remote Sensing Conference (LAGIRS), 2020, pp. 558–562.
[20] N. Fomferra, M. Böttcher, M. Zühlke, C. Brockmann, and E. Kwiatkowska, *Calvalus: Full-mission eo cal/val, processing and exploitation services*, in 2012 IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2012, pp. 5278–5281.
[21] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, et al., *Open mpi: Goals, concept, and design of a next generation mpi implementation*, in Recent Advances in Parallel Virtual Machine and Message Passing Interface: 11th European PVM/MPI Users' Group Meeting Budapest, Hungary, September 19-22, 2004. Proceedings 11, Springer, 2004, pp. 97–104.
[22] V. Garcia and M. M. Paulin, *Peps: Plateforme d'exploitation des produits sentinel*, in 2018 SpaceOps Conference, 2018, p. 2614.
[23] S. Gillies, B. Ward, A. Petersen, et al., *Rasterio: Geospatial raster i/o for python programmers*, URL https://github.com/mapbox/rasterio, (2013).
[24] G. Giuliani, B. Chatenoux, A. De Bono, D. Rodila, J.-P. Richard, K. Allenbach, H. Dao, and P. Peduzzi, *Building an Earth Observations Data Cube: Lessons learned from the Swiss Data Cube (SDC) on generating Analysis Ready Data (ARD)*, Big Earth Data, 1 (2017), pp. 100–117.
[25] G. Giuliani, J. Masó, P. Mazzetti, S. Nativi, and A. Zabala, *Paving the Way to Increased Interoperability of Earth Observations Data Cubes*, Data, 4 (2019), p. 113.
[26] V. C. Gomes, F. M. Carlos, G. R. Queiroz, K. R. Ferreira, and R. Santos, *Accessing and processing brazilian earth observation data cubes with the open data cube platform*, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 4 (2021), pp. 153–159.
[27] V. C. F. Gomes, G. R. Queiroz, and K. R. Ferreira, *An Overview of Platforms for Big Earth Observation Data Management and Analysis*, 12, p. 1253.
[28] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau, and R. Moore, *Google Earth Engine: Planetary-scale geospatial analysis for everyone*, Remote Sensing of Environment, 202 (2017), pp. 18–27.
[29] G. F. Guala, H. Hua, L. I. Duncanson, S. C. Niemoeller, N. Hunka, A. I. Mandel, and B. M. Freitag, *Biomass harmonization and sar analysis with the multi-mission algorithm and analysis platform (maap)*, in WGISS (Working Group on Information Systems and Services) 57th meeting, 2024.
[30] M. Hanson, *The open-source software ecosystem for leveraging public datasets in spatio-temporal asset catalogs (stac)*, in AGU Fall Meeting Abstracts, vol. 2019, 2019, pp. IN23B–07.
[31] I. A. T. Hashem, N. B. Anuar, A. Gani, I. Yaqoob, F. Xia, and S. U. Khan, *Mapreduce: Review and open challenges*, Scientometrics, 109 (2016), pp. 389–422.
[32] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, *Support vector machines*, IEEE Intelligent Systems and their applications, 13 (1998), pp. 18–28.
[33] S. Hoyer and J. Hamman, *xarray: N-D labeled arrays and datasets in Python*, Journal of Open Research Software, 5 (2017).
[34] I. Iosifescu Enescu, L. de Espona, D. Haas-Artho, R. Kurup Buchholz, D. Hanimann, M. Rüetschi, D. N. Karger, G.-K. Plattner, M. Hägeli, C. Ginzler, N. E. Zimmermann, and L. Pellissier, *Cloud Optimized Raster Encoding (CORE): A Web-Native Streamable Format for Large Environmental Time Series*, Geomatics, 1 (2021), pp. 369–382.
[35] A. Jacob, M. Mohr, P. J. Zellner, J. Dries, M. Claus, C. Briese, P. Griffitjs, and E. Pebesma, *Openeo platform brings analysis-ready data on demand*, in Proceedings of the 2021 conference on Big Data from Space: 18-20 May 2021, 2021, pp. 45–48.
[36] J. P. Jones, *Pbs: portable batch system*, (2001).
[37] I. Kamel and C. Faloutsos, *Hilbert r-tree: An improved rtree using fractals*, in VLDB, vol. 94, Citeseer, 1994, pp. 500–509.
[38] B. Killough, *Overview of the Open Data Cube Initiative*, in IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, 2018, pp. 8629–8632.
[39] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, and C. Willing, *Jupyter notebooks – a publishing format for reproducible computational workflows*, in Positioning and Power in Academic Publishing: Players, Agents and Agendas, F. Loizides and B. Schmidt, eds., IOS Press, 2016, pp. 87–90.

[40] S. Kopp, P. Becker, A. Doshi, D. J. Wright, K. Zhang, and H. Xu, *Achieving the Full Vision of Earth Observation Data Cubes*, Data, 4 (2019), p. 94.
[41] S. Kothari, J. Shah, J. Verma, S. H. Mankad, and S. Garg, *Raster big data processing using spark with geotrellis*, in International Conference on Computing, Communication and Learning, Springer, 2023, pp. 260–271.
[42] M. Krämer, R. Gutbell, H. M. Würz, and J. Weil, *Scalable processing of massive geodata in the cloud: Generating a level-of-detail structure optimized for web visualization*, AGILE: GIScience Series, 1 (2020), pp. 1–20.
[43] A. Lewis, J. Lacey, S. Mecklenburg, J. Ross, A. Siqueira, B. Killough, Z. Szantoi, T. Tadono, A. Rosenavist, P. Goryl, N. Miranda, and S. Hosford, *Ceos analysis ready data for land (card4l) overview*, in IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, 2018, pp. 7407–7410.
[44] A. Lewis, L. Lymburner, M. B. J. Purss, B. Brooke, B. Evans, A. Ip, A. G. Dekker, J. R. Irons, S. Minchin, N. Mueller, S. Oliver, D. Roberts, B. Ryan, M. Thankappan, R. Woodcock, and L. Wyborn, *Rapid, high-resolution detection of environmental change over continental scales from satellite data – the Earth Observation Data Cube*, International Journal of Digital Earth, 9 (2016), pp. 106–111.
[45] A. Lewis, S. Oliver, L. Lymburner, B. Evans, L. Wyborn, N. Mueller, G. Raevksi, J. Hooke, R. Woodcock, J. Sixsmith, W. Wu, P. Tan, F. Li, B. Killough, S. Minchin, D. Roberts, D. Ayers, B. Bala, J. Dwyer, A. Dekker, T. Dhu, A. Hicks, A. Ip, M. Purss, C. Richards, S. Sagar, C. Trenham, P. Wang, and L.-W. Wang, *The Australian Geoscience Data Cube — Foundations and lessons learned*, Remote Sensing of Environment, 202 (2017), pp. 276–292.
[46] Z. Li, *Geospatial Big Data Handling with High Performance Computing: Current Approaches and Future Directions*, in High Performance Computing for Geospatial Applications, W. Tang and S. Wang, eds., Springer International Publishing, 2020, pp. 53–76.
[47] V. Lonjou, C. Desjardins, O. Hagolle, B. Petrucci, T. Tremas, M. Dejus, A. Makarau, and S. Auer, *MACCS-ATCOR joint algorithm (MAJA)*, in Remote Sensing of Clouds and the Atmosphere XXI, vol. 10001, SPIE, 2016, pp. 25–37.
[48] S. S. Mahammad and R. Ramakrishnan, *Geotiff-a standard image file format for gis applications*, Map India, (2003), pp. 28–31.
[49] M. D. Mahecha, F. Gans, G. Brandt, R. Christiansen, S. E. Cornell, N. Fomferra, G. Kraemer, J. Peters, P. Bodesheim, G. Camps-Valls, J. F. Donges, W. Dorigo, L. M. Estupinan-Suarez, V. H. Gutierrez-Velez, M. Gutwin, M. Jung, M. C. Londoño, D. G. Miralles, P. Papastefanou, and M. Reichstein, *Earth system data cubes unravel global multivariate dynamics*, Earth System Dynamics, 11 (2020), pp. 201–234.
[50] M. Main-Knorn, B. Pflug, J. Louis, V. Debaecker, U. Müller-Wilm, and F. Gascon, *Sen2Cor for sentinel-2*, in Image and Signal Processing for Remote Sensing XXIII, vol. 10427, International Society for Optics and Photonics, p. 1042704.
[51] P. Mehrotra, J. Djomehri, S. Heistand, R. Hood, H. Jin, A. Lazanoff, S. Saini, and R. Biswas, *Performance evaluation of amazon ec2 for nasa hpc applications*, in Proceedings of the 3rd workshop on Scientific Cloud Computing, 2012, pp. 41–50.
[52] D. Merkel, *Docker: lightweight linux containers for consistent development and deployment*, Linux journal, 2014 (2014), p. 2.
[53] M. O. Mete, *Geospatial big data analytics for sustainable smart cities*, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 48 (2023), pp. 141–146.
[54] M. O. Mete and T. Yomralioglu, *Implementation of serverless cloud GIS platform for land valuation*, International Journal of Digital Earth, 14 (2021), pp. 836–850.
[55] O. S. Microsoft, M. McFarland, R. Emanuele, D. Morris, and T. Augspurger, *Microsoft/PlanetaryComputer: October 2022*.
[56] G. Milcinski, J. Bojanowski, D. Clarijs, and J. de la Mar, *Copernicus Data Space Ecosystem - Platform That Enables Federated Earth Observation Services and Applications*, in IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium, 2024, pp. 875–877.
[57] A. Miles, J. Kirkham, M. Durant, J. Bourbeau, T. Onalan, J. Hamman, Z. Patel, shikharsg, M. Rocklin, raphael dussin, V. Schut, E. S. de Andrade, R. Abernathey, C. Noyes, sbalmer, pyup.io bot, T. Tran, S. Saalfeld, J. Swaney, J. Moore, J. Jevnik, J. Kelleher, J. Funke, G. Sakkis, C. Barnes, and A. Banihirwe, *zarr-developers/zarr-python: v2.4.0*, Apr. 2020.
[58] M. Mueller and B. Pross, *Ogc wps 2.0 interface standard. version 2.0.*, Open Geospatial Consortium, (2015).
[59] N. Mueller, A. Lewis, D. Roberts, S. Ring, R. Melrose, J. Sixsmith, L. Lymburner, A. McIntyre, P. Tan, S. Curnow, and A. Ip, *Water observations from space: Mapping surface water from 25years of Landsat imagery across Australia*, Remote Sensing of Environment, 174 (2016), pp. 341–352.
[60] J. Musial, J. Leszczenski, J. Bojanowski, G. Milcinski, A. Vrecko, D. Clarijs, J. Dries, and U. Marquard, *Overview of the Copernicus Data Space Ecosystem APIs*.
[61] V. Naeimi, S. Elefante, S. Cao, W. Wagner, A. Dostalova, and B. Bauer-Marschallinger, *Geophysical parameters retrieval from sentinel-1 sar data: a case study for high performance computing at eodc*, in Proceedings of the 24th High Performance Computing Symposium, 2016, pp. 1–8.
[62] M. Neagul, I. Nedelcu, and A. Munteanu, *Building a national spatio-temporal datacube*, in IGARSS 2023-2023 IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2023, pp. 5089–5092.
[63] T. E. Odaka, A. Banihirwe, G. Eynard-Bontemps, A. Ponte, G. Maze, K. Paul, J. Baker, and R. Abernathey, *The pangeo ecosystem: Interactive computing tools for the geosciences: Benchmarking on hpc*, in Tools and Techniques for High Performance Computing: Selected Workshops, HUST, SE-HER and WIHPC, Held in Conjunction with SC 2019, Denver, CO, USA, November 17–18, 2019, Revised Selected Papers 6, Springer, 2020, pp. 190–204.
[64] M. C. A. Picoli, R. Simoes, M. Chaves, L. A. Santos, A. Sanchez, A. Soares, I. D. Sanches, K. R. Ferreira, and G. R.

Queiroz, *CBERS DATA CUBE: A POWERFUL TECHNOLOGY FOR MAPPING AND MONITORING BRAZILIAN BIOMES*, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, V-3-2020 (2020), pp. 533–539.

[65] R. Rew and G. Davis, *Netcdf: an interface for scientific data access*, IEEE computer graphics and applications, 10 (1990), pp. 76–82.

[66] O. Ronneberger, P. Fischer, and T. Brox, *U-net: Convolutional networks for biomedical image segmentation*, in Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18, Springer, 2015, pp. 234–241.

[67] M. Saeedan and A. Eldawy, *Spatial parquet: a column file format for geospatial data lakes*, in Proceedings of the 30th International Conference on Advances in Geographic Information Systems, 2022, pp. 1–4.

[68] R. Schnabel and R. Klein, *Octree-based point-cloud compression.*, PBG@ SIGGRAPH, 3 (2006), pp. 111–121.

[69] A. W. Services, *Amazon elastic compute cloud (ec2)*. `https://aws.amazon.com/ec2/`, 2024. Accessed: 2024-04-21.

[70] R. Simoes, G. Camara, G. Queiroz, F. Souza, P. R. Andrade, L. Santos, A. Carvalho, and K. Ferreira, *Satellite Image Time Series Analysis for Big Earth Observation Data*, Remote Sensing, 13 (2021), p. 2428.

[71] STAC Contributors, *SpatioTemporal asset catalog (STAC) specification*.

[72] T. Storch, C. Reck, S. Holzwarth, and V. Keuck, *Code-de-the german operational environment for accessing and processing copernicus sentinel products*, in IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium, IEEE, 2018, pp. 6520–6523.

[73] T. Storch, C. Reck, S. Holzwarth, B. Wiegers, N. Mandery, U. Raape, C. Strobl, R. Volkmann, M. Böttcher, A. Hirner, et al., *Insights into code-de–germany's copernicus data and exploitation platform*, Big Earth Data, 3 (2019), pp. 338–361.

[74] M. Sudmanns, H. Augustin, B. Killough, G. Giuliani, D. Tiede, A. Leith, F. Yuan, and A. Lewis, *Think global, cube local: An Earth Observation Data Cube's contribution to the Digital Earth vision*, Big Earth Data, 0 (2022-07-21), pp. 1–29.

[75] M. Sudmanns, H. Augustin, L. van der Meer, A. Baraldi, and D. Tiede, *The Austrian Semantic EO Data Cube Infrastructure*, Remote Sensing, 13 (2021), p. 4807.

[76] M. Sudmanns, D. Tiede, S. Lang, H. Bergstedt, G. Trost, H. Augustin, A. Baraldi, and T. Blaschke, *Big Earth data: Disruptive changes in Earth observation data management and analysis?*, International Journal of Digital Earth, 13 (2020), pp. 832–850.

[77] D. Thain, T. Tannenbaum, and M. Livny, *Distributed computing in practice: the condor experience.*, Concurrency - Practice and Experience, 17 (2005), pp. 323–356.

[78] M. Tremmel, *COMTILES: A CASE STUDY OF A CLOUD OPTIMIZED TILE ARCHIVE FORMAT FOR DEPLOYING PLANET-SCALE TILSETS IN THE CLOUD*, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLVIII-4-W7-2023 (2023-06-22), pp. 231–237.

[79] A. Wachs and E. T. Zacharatou, *Analysis of geospatial data loading*, in Proceedings of the Tenth International Workshop on Testing Database Systems, 2024, pp. 36–42.

[80] C. Xu, X. Du, H. Jian, Y. Dong, W. Qin, H. Mu, Z. Yan, J. Zhu, and X. Fan, *Analyzing large-scale Data Cubes with user-defined algorithms: A cloud-native approach*, International Journal of Applied Earth Observation and Geoinformation, 109 (2022), p. 102784.

[81] C. Xu, X. Du, Z. Yan, and X. Fan, *Scienceearth: A big data platform for remote sensing data processing*, Remote Sensing, 12 (2020), p. 607.

[82] D. Xu, Y. Ma, J. Yan, P. Liu, and L. Chen, *Spatial-feature data cube for spatiotemporal remote sensing data processing and analysis*, Computing, 102 (2020), pp. 1447–1461.

[83] C. Yang, M. Yu, F. Hu, Y. Jiang, and Y. Li, *Utilizing Cloud Computing to address big geospatial data challenges*, Computers, Environment and Urban Systems, 61 (2017), pp. 120–128.

[84] A. B. Yoo, M. A. Jette, and M. Grondona, *Slurm: Simple linux utility for resource management*, in Workshop on job scheduling strategies for parallel processing, Springer, 2003, pp. 44–60.

[85] J. Yu, J. Wu, and M. Sarwat, *Geospark: A cluster computing framework for processing large-scale spatial data*, in Proceedings of the 23rd SIGSPATIAL international conference on advances in geographic information systems, 2015, pp. 1–4.

[86] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, et al., *Apache spark: a unified engine for big data processing*, Communications of the ACM, 59 (2016), pp. 56–65.

[87] C. Zhang, L. Di, Z. Sun, G. Y. Eugene, L. Hu, L. Lin, J. Tang, and M. S. Rahman, *Integrating ogc web processing service with cloud computing environment for earth observation data*, in 2017 6th International Conference on Agro-Geoinformatics, IEEE, 2017, pp. 1–4.