AGU ADVANCING EARTH AND SPACE SCIENCES

Earth and Space Science

RESEARCH ARTICLE
10.1029/2025EA004251

Collaborative and Reproducible Planetary Science Through the Europlanet GMAP JupyterHub Processing Environment

G. Nodjoumi[1,2,3], C. H. Brandt[4], J. E. Suárez-Valencia[1], E. Luzzi[5], M. Valiante[6], and A. P. Rossi[7]

[1]School of Science, Constructor University Bremen gGmbH, Bremen, Germany, [2]Space Science Data Center (SSDC), Agenzia Spaziale Italiana (ASI), Via del Politecnico snc, Rome, Italy, [3]INAF/Osservatorio Astronomico di Roma (INAF-OAR), Monte Porzio Catone, Italy, [4]EGI Foundation, Amsterdam, The Netherlands, [5]Mississippi Mineral Resources Institute, University of Mississippi, University, MS, USA, [6]Dipartimento di Ingegneria Civile, Università degli Studi di Salerno, Fisciano, Italy, [7]Earthgraph GmbH, Bremen, Germany

Key Points:

- JupyterHub facilitates multi-user access to computational resources and promotes reproducibility in data analysis
- A Docker-based USGS Integrated Software for Imagers and Spectrometers and the NASA Ames Stereo Pipeline, allows efficient geospatial and stereogrammetric processing of planetary data
- The Europlanet GMAP initiative promotes FAIR practices in planetary science via open-source platforms and training programs

Correspondence to:
G. Nodjoumi,
gnodjoumi@constructor.university

Citation:
Nodjoumi, G., Brandt, C. H., Suárez-Valencia, J. E., Luzzi, E., Valiante, M., & Rossi, A. P. (2025). Collaborative and reproducible planetary science through the Europlanet GMAP JupyterHub processing environment. *Earth and Space Science*, *12*, e2025EA004251. https://doi.org/10.1029/2025EA004251

Received 4 FEB 2025
Accepted 28 APR 2025

Author Contributions:
Conceptualization: G. Nodjoumi, C. H. Brandt, A. P. Rossi
Formal analysis: G. Nodjoumi, J. E. Suárez-Valencia, E. Luzzi, M. Valiante
Investigation: G. Nodjoumi, C. H. Brandt, J. E. Suárez-Valencia, E. Luzzi, M. Valiante
Methodology: G. Nodjoumi, C. H. Brandt
Project administration: A. P. Rossi
Software: G. Nodjoumi, C. H. Brandt
Supervision: A. P. Rossi
Validation: G. Nodjoumi, C. H. Brandt
Visualization: G. Nodjoumi
Writing – original draft: G. Nodjoumi
Writing – review & editing: C. H. Brandt, J. E. Suárez-Valencia, E. Luzzi, M. Valiante, A. P. Rossi

Abstract JupyterHub is an open-source system enabling multiple users to access individual computational environments. This facilitates collaborative development and execution of Jupyter notebooks, Python scripts, and other tools among researchers and educators through a unified interface. Through the integration of container technologies, including Docker, JupyterHub achieves seamless scalability for numerous users while maintaining efficient computational resource management. This flexible approach is especially useful in specialized areas like planetary data science, which requires robust and reproducible workflows to manage large volumes of mission data. The Europlanet Geologic MApping of Planetary surfaces (GMAP) project employs a Docker-based JupyterHub deployment to centralize essential data processing tools, such as the Integrated Software for Imagers and Spectrometers (ISIS) and the NASA Ames Stereo Pipeline (ASP). These open-source tools facilitate tasks ranging from image calibration and map projection to stereogrammetry and 3D modeling. The deployment of these elements within Docker containers facilitates simplified installation and consistent performance across disparate hardware configurations. The use of pre-configured image formats within ISIS, ASP, and other GIS and Python libraries allows planetary scientists to efficiently process raw data into analytical products, including Digital Terrain Models. Additionally, JupyterHub's architecture enables secure collaboration via authentication methods (e.g., OAuth, GitHub), with concurrent provision for private and shared data directories. This integrated framework promotes reproducible research by streamlining the sharing of scripts, notebooks, and workflows. The GMAP JupyterHub platform significantly accelerates scientific discovery through the reduction of technical barriers, the promotion of standardization, and the provision of global access to planetary data science resources.

Plain Language Summary A shared web interface on JupyterHub simplifies collaborative coding for multiple users. To avoid individual software installations, JupyterHub leverages Docker containers to package necessary programs and libraries, enabling scientists to use a consistent setup through a web browser. The Europlanet Geologic MApping of Planetary surfaces (GMAP) project uses this setup to streamline the processing of space mission data from Mars, the Moon, and other celestial bodies. Researchers can directly utilize key tools like Integrated Software for Imagers and Spectrometers and the NASA Ames Stereo Pipeline within Jupyter notebooks for image processing and 3D map/model creation. Because all essential parts are bundled in Docker images, these tools function seamlessly. This approach avoids installation problems, saves time, and standardizes the work environment for all users of these containers. The integration of these tools in a shared online space makes data, script, and result sharing easier for teams. Consistent research is encouraged, and data processing is simplified for even novice users. With its container-based design, GMAP's JupyterHub setup allows scientists to prioritize planetary scientific discoveries over software configuration complexities.

1. Introduction

1.1. Overview of JupyterHub

JupyterHub is an open-source platform, developed by Project Jupyter, designed to provide access to local or remote computational resources through a JupyterLab interface. JupyterHub offers a centralized environment where multiple users can log in simultaneously to personal JupyterLab instances with dedicated and shared resources (Beg et al., 2021; Granger & Pérez, 2021; Kluyver et al., 2016). Each user can create and execute Jupyter

notebooks, python and bash scripts, as well as create python virtual environments. Initially developed to streamline research, education, and data science collaborations, JupyterHub provides a flexible environment where users can run and share live code, equations, visualizations, and narrative text within a single, interactive document.

Researchers, educators, and technical users benefit from JupyterHub's interactive, collaborative computing environments. This resource is designed for scientists, educators, students, and data analysts, especially those specializing in planetary science, data-intensive research, or geospatial analysis. JupyterHub uses Docker images that offer a broad range of programming languages and tools, even older versions, all delivered through a JupyterLab interface. This design accommodates both beginning users who appreciate ease of use and advanced users who demand extensive customization and powerful computing. Such flexibility enables planetary scientists to conveniently alternate between different analytical tools within a single platform. The configuration allows for the utilization of diverse authentication protocols (including remote OAuth servers such as GitHub), enabling user access management and ensuring the protection of shared resources, with enhanced flexibility and scalability. When using container technologies such as Docker (Merkel, 2014), JupyterHub can effectively handle scalability in cloud and distributed computational resources and, the docker images provided through the Hub can be shared and used independently.

The adaptability, scalability, and simplicity of JupyterHub render it a fitting choice for the Europlanet GMAP project (Nass et al., 2021). The project's goal is to enhance planetary data science through a centralized platform for data analysis and teamwork.

1.2. Docker-Based Deployments

Docker is a technology that enables the execution of software applications within isolated environments known as containers (Merkel, 2014). Each container includes all necessary dependencies, libraries, and configurations, making it easy to deploy complex tools without conflicts or installation hurdles.

Docker ensures isolation for each user's work environment to prevent conflicts with dependencies and allows compatibility with shared data. Despite the isolation provided by containerization, security risks persist, mainly when Docker containers run with root privileges, which is the default setting. To reduce these risks, we have tested and verified our deployment using rootless Podman (Heon et al., 2018), improving security by preventing containers from running with elevated privileges on the host. For enhanced system security, implement strict firewall rules that block access to insecure or unneeded ports, thereby reducing the attack surface. Strengthening system security requires careful management of user permissions in filesystems and containerized environments by utilizing Access Control Lists (ACLs). With ACLs, administrators have granular control over files and directories, defining specific read, write, and execute permissions for users and groups, exceeding the capabilities of standard Unix permissions. ACLs, when paired with containerization, secure sensitive data by isolating user environments and ensuring compliance with institutional policies against unauthorized access and data breaches. Only authenticated users, through secure, institutionally and trusted managed identity providers, should be able to access JupyterHub. For example, eduGAIN integration allows authentication via trusted academic identity federations, thereby ensuring proper user verification. A robust, reliable logging system is necessary for monitoring and incident response. This system must record all system and user activity and comply with privacy regulations. Access control configurations, container privilege management, and authentication systems are highly infrastructure-dependent and require thorough, dedicated examination. Therefore, they are beyond the scope of this work.

Planetary scientists have the flexibility to transition between tools such as ISIS and ASP without concerns about system-wide compatibility. Docker containers ensure consistent performance of code across various platforms, a critical requirement for reproducibility in scientific studies.

Docker containers are efficient and lightweight, enabling JupyterHub to efficiently accommodate many concurrent users. The deployment of Docker containers across different infrastructures allows planetary scientists to run their workflows in a versatile manner, spanning from local machines to cloud environments. For enhanced scalability and orchestration, incorporating tools like Kubernetes (Brewer, 2015; Kubernetes, 2025) and OpenShift (Red Hat, Inc, 2025) can be beneficial. Kubernetes offers robust automation for deploying, scaling, and managing application containers across multiple host clusters. OpenShift, as well as Kubernetes, provides

advanced enterprise capabilities, such as heightened security and optimized workflows. The integration of JupyterHub and these orchestration tools facilitates the efficient management of Jupyter notebooks across a scalable and secure infrastructure.

1.3. Integrated Software for Imagers and Spectrometers (ISIS)

ISIS is one of the essential tools integrated into this Dockerized JupyterHub setup. Developed by the USGS Astrogeology Science Center, ISIS is an open-source software suite crafted for processing planetary data obtained from space missions (Sucharski et al., 2020). This tool is indispensable for producing analysis-ready products, and other scientific outcomes. Basic image processing tools are integrated within ISIS, such as contrast adjustments, image algebra, and statistical analysis. Additionally, it provides map-projection support for a broad spectrum of missions, including NASA's Apollo, Voyager, Cassini, Mars Reconnaissance Orbiter (MRO), and international missions like Chandrayaan and Kaguya. In the *Europlanet GMAP* deployment, the pre-configured ISIS Docker images enable planetary scientists to process and analyze mission data efficiently, with no need for time-consuming software setups. This allows researchers to focus on their data processing, from raw data to map-projected imagery and Digital Terrain Models (DTMs), and on the interpretation of such data, without worrying about software installations and setup.

1.4. NASA Ames Stereo Pipeline (ASP)

ASP is another integral tool in the GMAP Dockerized JupyterHub. ASP is a suite of open-source geodesy and stereogrammetry tools that are used to process satellite imagery, rover images, and aerial camera data into cartographic products such as DTMs, ortho-projected images, and 3D models (Beyer et al., 2018). As stated above, ASP core algorithms are developed around two main techniques, Stereogrammetry and Structure-from-Motion and several highly customizable configurations and algorithms are provided for each of these techniques. Within the framework of the GMAP project, ASP's Docker integration enables planetary scientists to efficiently analyze vast data sets and generate essential cartographic materials.

Both ISIS and ASP are open-source tools, released and maintained periodically. Their usage is relatively simple, although the installation process may be challenging due to various prerequisites that need to be fulfilled for the operating system and package dependencies.

1.5. Europlanet GMAP Project

The Europlanet Geologic MApping of Planetary surfaces (GMAP) initiative is focused on producing precise, uniform geological maps of celestial bodies in our Solar System. The Europlanet 2024 Research Infrastructure (RI) project, funded by the European Union, aims to facilitate collaborative planetary research and offer resources to scientists worldwide. GMAP specifically addresses the need for geospatial and geological information that supports scientific exploration and analysis of planetary surfaces, ranging from terrestrial planets to the moons of the outer planets.

GMAP adheres to international standards to guarantee the consistency and global usability of maps by scientists. The project's maps are developed in accordance with protocols specified by the United States Geological Survey (USGS) and other geospatial organizations (Federal Geographic Data Committee, 1998; Open Geospatial Consortium, 2019), aligning data presentation with globally accepted mapping frameworks. The main goal is to map the surfaces of a range of planetary bodies, including Mars, the Moon, Venus, Mercury, and outer Solar System moons such as Europa and Enceladus. The process of mapping aids in the identification of geological characteristics like craters, valleys, ridges, and possible volcanic occurrences. One objective of the project is to ensure the accessibility of these maps to a wide range of people. GMAP combines data from diverse missions and sources, including data from ESA, NASA, and other space agencies. These maps, which are produced using GIS tools and software to study and display planetary surfaces, are open to both researchers and the public, serving as a crucial resource for education and research in planetary science. The development of maps involves the utilization of tools for 3D modeling, spatial analysis, and high-resolution imagery, thus enabling thorough exploration of planetary geology. The detailed maps created by GMAP are fundamental tools for upcoming space missions, assisting in the selection of landing sites, evaluation of hazards, and identification of potential resources. This holds particular significance for missions to Mars and the Moon, as knowledge of surface conditions is essential for human and robotic exploration planning.

GMAP's purpose extends beyond map production. The project provides educational programs and workshops, such as the GMAP Winter School in its fourth edition to date, to assist scientists and students in comprehending planetary geological mapping techniques and standards. These training initiatives are essential in building a skilled workforce in planetary science.

Europlanet GMAP is crucial for advancing our comprehension of planetary geology through the provision of top-notch, standardized, and readily available geological maps. These resources not only support current scientific research but also pave the way for future exploration and discovery in planetary science.

1.6. Motivation

Before delving into scientific analysis of planetary data, beginners in the field of planetary science often find it necessary to first undertake the preliminary task of data processing. Despite the existence of multiple data archives, like the Orbital Data Explorer hosted by the Planetary Data System (PDS Geosciences Nodes, 2020), containing vast quantities of advanced products, additional processing is required in many instances, particularly for higher-level data like map-projected images or DTMs.

As previously mentioned, the conventional workflows necessitate the utilization of USGS ISIS. Yet, the installation procedure could pose difficulties for scientists with limited computer system and coding skills, and this applies for ASP as well. Additionally, it is crucial to efficiently handle multiple versions to ensure the reproducibility and dissemination of ongoing research due to the possibility of various versions of ISIS and ASP being used in published public workflows, scripts, and code. Favorably, a number of online platforms provide access to advanced products that are pre-processed and ready for analysis, whereas others provide some level of processing capabilities. For example, MarsSI is an online platform that offers access to High-Resolution Imaging Science Experiment (HiRISE) and CTX map-projection and DTMs generation using ASP, whereas platforms such as NASA Treks, QuickMap, JMars, and MATISSE provide limited processing capabilities but allow access to various high-level products from different missions.

Several alternatives to JupyterHub exist, including platforms such as Google Colab (Bisong, 2019) and Binder (Jupyter et al., 2018). Despite offering complimentary service with limited resources and paid enhancements, these platforms exhibit key distinctions from JupyterHub. For organizational use, JupyterHub provides an open-source solution for the deployment and administration of Jupyter Notebook environments supporting multiple users. By contrast, Google Colaboratory is a proprietary cloud service provided by Google, whereas Binder is an open-source service that enables users to create and share interactive computing environments. The Binder project, much like our GMAP JupyterHub, aims to provide a simple platform for sharing computing environments (Jupyter et al., 2018). It allows users to create customized environments and distribute them using a unique link. Utilization scenarios include workshops, scientific workflows, and enhancing collaboration among teams. These scenarios utilize repo2docker (Forde et al., 2018) to generate docker images from GitHub repositories, following established guidelines. The images are housed in an online repository, allowing Binder to create a collaborative environment that is open to all. Regrettably, the approach cannot be fully suitable due to the necessity of ISISDATA for both ISIS and ASP, which contains important data such as mission kernels, camera models, and orbital parameters. Data from ongoing missions contributes to the growth of the ISISDATA repository, which now exceeds 2 TB. The substantial size of this volume poses considerable challenges for institutions with limited storage or network capacity. Centralized, remote access to this data set enables users to perform analyses and workflows directly through the online platform, eliminating the need for local data downloads or hosting. The infrastructure requirements for end-users are significantly lessened by this approach, resulting in increased platform accessibility for a broader global research community. Furthermore, planetary data processing tasks demand significant computational resources and can take hours (if not days) to finish, making it unsuitable for the platforms mentioned above.

In this scenario, we have decided to develop a new platform that offers complete processing capabilities without requiring complex installation procedures and the ability to be deployed without restrictions both online and offline.

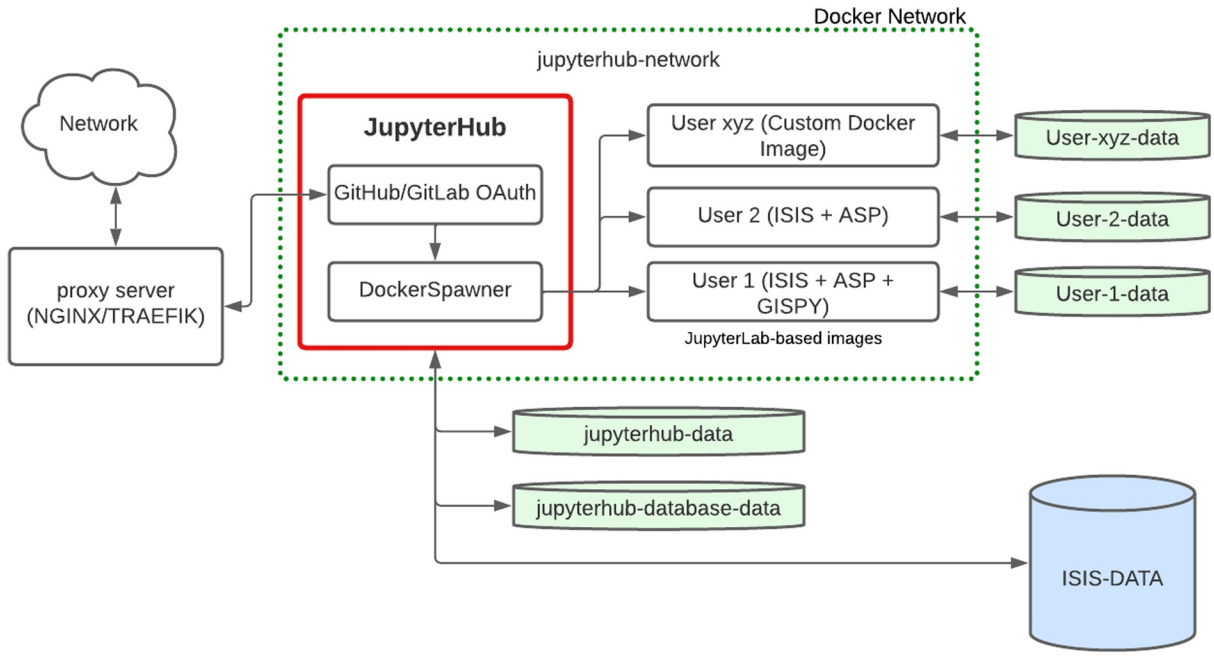

Figure 1. Flowchart of the Geologic MApping of Planetary surfaces (GMAP) JupyterHub Architecture. The core JupyterHub components, namely the authentication system and the DockerSpawner module, are highlighted in red. The green-dotted area represents the GMAP JupyterHub deployment, which is isolated within a dedicated Docker network. In this environment, user-specific Docker containers (ISIS5, ASP3, GISPY, and JupyterLab) are instantiated and linked to persistent data volumes (cylindrical storage). This setup ensures modularity, environment isolation, and secure access to shared planetary data sets.

2. Europlanet GMAP JupyterHub

To simplify the complex processing tasks for end users, we began creating customizable Docker images that integrate ISIS, ASP, and other commonly used packages in planetary sciences, including tools for reading and plotting final products. The GMAP Docker-based JupyterHub setup (Figure 1) offers a sturdy and adaptable environment tailored for deploying JupyterHub with advanced customization choices to meet varied user requirements in a collaborative computing setting. By utilizing Docker containers, the deployment and maintenance of JupyterLab instances are strongly simplified, positioning it as a valuable solution for both individual and team-oriented workflows.

We selected GitHub as our primary platform due to its free access and ease of implementation. Integration testing was performed with our self-hosted GitLab instance, taking advantage of its robust self-hosting functionality. This configuration enables users to securely access their accounts on these platforms, streamlining user onboarding through popular identity providers, minimizing the reliance on unique credentials. Implementing OAUth authentication improves security and user convenience with single sign-on, ensuring easy and secure access within a unified development environment. Additional authentication methods can be enabled through JupyterHub settings. The recent deployment of a self-hosted Authentik (Authentik Security Inc, 2025) identity provider has expanded our infrastructure and improved the flexibility of our authentication processes, leveraging an open-source solution.

The Docker-based setup enables the utilization of various JupyterLab environments by utilizing distinct Docker images tailored to different ISIS and ASP versions or variants, along with supplementary tools like DeepLandforms (Nodjoumi, Pozzobon, et al., 2023) and MoonIndex (Suárez-Valencia et al., 2024) from Europlanet GMAP, facilitating access to multiple pre-configured environments concurrently. Users have the option to select the environment that best fits their project requirements by customizing each Docker image with specific tools, libraries, and settings, like data science, GIS, or image processing. By adopting a multi-environment approach, such as deploying user-specific containers, resource utilization can be optimized through on-demand provisioning and scalable resource allocation. Although operating multiple Docker environments introduces a degree of computational and storage overhead relative to a singular environment, the advantages are realized through enhanced isolation, modularity, and scalability. Containerization allows customized configurations to meet

individual user requirements, including only essential dependencies and services; this approach mitigates the inefficiencies inherent in monolithic, generalized systems. Furthermore, idle environments may be paused or terminated to mitigate unnecessary resource consumption. The flexibility inherent in this system simplifies management and ensures the precise utilization of computational and storage resources, thus improving overall infrastructural efficiency and maintainability. The configuration offers effective storage management choices for managing both private and shared data. User-specific data is stored in individual directories, creating a secure environment separated from other users. Conventional ACL systems permit granular control over directory and file access, assigning specific permissions to individual users or groups for each operation (reading, writing, and executing). This approach allows for granular control over access rights on a per-object basis, enabling multiple users to share access to the same resources with different permission levels. Alternatively, implementing individual directories for user-specific data storage affords each user a dedicated and exclusively accessible directory. This method facilitates permission management by intrinsically isolating user data, thus negating the requirement for elaborate ACL configurations. Although this approach strengthens security by isolating data, it may impede collaborative work requiring shared access to specific files or directories because each user's data is contained separately. Furthermore, shared data storage enables collaborative projects by granting access to data and resources among various user accounts or projects. The differentiation between private and shared storage upholds data security and improves team-based project management features within a centrally managed framework. Moreover, we integrated JupyterHub with a self-hosted instance of Nextcloud (Karlitschek, 2018) a third-party cloud storage provider.

The initialization scripts are set up to execute before a user log in, configuring their environment by handling tasks like setting file paths, environment variables, and loading resources. Administrators have the ability to customize pre-login scripts to set default configurations, libraries, and resources, thereby streamlining manual configuration processes. The presence of this function is crucial for establishing a coherent and standardized user experience, ensuring that all users commence with a suitably configured operational environment. This particular configuration is a prerequisite for accessing ISIS-ASP docker images through JupyterHub but not required for standalone usage.

2.1. Docker Images for GMAP

The setup includes specially developed Docker images for the geospatial data and mapping application workflows:

- *jupyter-gispy*: A Docker image tailored for GIS and Python integration, offering extensive geospatial analysis capabilities. This environment is well-suited for general geospatial data science tasks, utilizing Python-based GIS libraries.
- *jupyter-isis*: Docker image dedicated to the ISIS toolset, providing users with powerful geospatial analysis tools for general planetary science.
- *jupyter-isis-asp*: A Docker image combining ISIS and ASP, suited for tasks involving planetary and stereo imaging.
- *jupyter-grass*: A Docker image featuring JupyterLab preconfigured with GRASS GIS, enabling advanced geospatial data processing, raster/vector analysis, and environmental modeling directly within the notebook interface.
- *jupyter-grass-gispy*: A comprehensive Docker environment that combines GRASS GIS with Python-based GIS tools. Users who need GRASS GIS's native geospatial functionality and the flexibility offered by Python geospatial libraries will find this hybrid setup ideal.
- *jupyter-fortran*: scientific computing and legacy code integration via a JupyterLab environment, complete with a Fortran compiler and necessary tools, supporting a modern notebook workflow.

These images are designed to meet the needs of GMAP users, providing optimized, isolated environments for high-performance computing in geospatial applications. The Docker images utilized in this arrangement are specifically crafted for autonomous operation, allowing them to function independently in any Docker environment without the necessity of the entire JupyterHub setup. The publication of each image on DockerHub and GitHub guarantees accessibility, transparency, and ongoing updates. Users have the capability to retrieve the most recent versions directly from DockerHub, facilitating efficient deployment and seamless updates. GitHub

repositories offer Dockerfiles and documentation, supporting users in understanding the configuration of each image and customizing them if required.

3. Use Cases and Applications

During the late stages of the development, and right after the deployment of the Europlanet GMAP JupyterHub, we tested its versatility and usability across several use cases. From python-based code development to planetary data processing, and planetary workshops.

3.1. Planetary Data Processing

Europlanet GMAP JupyterHub was used as an online platform to convert planetary data into map-projected COG-compliant GeoTIFF files. We tested several data types from different missions, including the Lunar Reconnaissance Orbiter Cameras (LROC) Wide-Angle Camera (WAC) and Narrow Angle Camera (NAC), the MRO Context Camera (CTX) and HiRISE, and the Chandraayan-1 Moon Mineralogy Mapper (M3). LROC NAC/WAC and MRO HiRISE/CTX data sets were processed to obtain high-resolution images and mosaics. Stereo acquisitions were then processed further with the NASA ASP tool, to obtain high-resolution DTMs. LROC data were used mainly for the EXPLORE project (Cox et al., 2022; Nodjoumi et al., 2022) and to prepare the materials for the EXPLORE's Data challenges (Lovell et al., 2024; Nodjoumi, Valencia, et al., 2023). MRO data were used to study several areas of Mars, including landing site characterization for near-surface ice content (Luzzi, Heldmann, et al., 2023), characterization of lava plain sequences (Characterization of a lava plain NW of Ascraeus Mons, Mars, through surface morphometric analyses and SHARAD subsurface detections.), and characterization of lunar dome surfaces, conducted by incorporating LROC data with M3 data (Suárez-Valencia et al., 2024).

3.2. Open-Source Tools Development

A Semiautomated Analysis of Displacement-To-Length Scaling of the Grabens Affecting Lunar Floor-Fractured Craters (Luzzi, Nodjoumi, et al., 2023).

We developed a python-based approach implemented through a series of Jupyter Notebooks, which were designed to automate a process typically done manually in GIS software. The workflow comprises the following tasks:

- Fault Mapping: Individual faults were manually mapped as shapefiles in QGIS. This stage utilized a specific scale based on fault size and supplemented imaging with slope maps for better fault visibility.
- Vertex Densification: Because manually mapped faults had irregularly spaced vertices, the code replaced them with equally spaced vertices for consistent analysis.
- Graben Width Calculation: Using the vertices, the width of each graben was determined by calculating the minimum distance between opposite fault vertices.
- Transect Generation: For each vertex, transects (cross-sections) were created to measure fault displacement across the graben.
- Topographic Profiling: DTMs combined with transect geometry provided detailed topographic profiles, used to calculate fault displacement using a 3D distance formula.
- Data Aggregation and Scaling: Statistics on fault length, displacement, and width were compiled. Data were normalized and scaled using Python's scikit-learn for comparative analysis across different faults and craters.

MoonIndex, an Open-Source Tool to Generate Spectral Indexes for the Moon From M3 Data (Suárez-Valencia et al., 2024).

We developed a python-based tool to process map-projected M3 data cubes, with the aim of reducing the noise, remove the spectral continuum, and generate spectral indexes. This tool recreates the spectral indexes defined by previous authors over the years, compiling them in an open-source python package. The overall workflow involves several steps:

- Data Preparation: M3 cubes are first pre-processed using ISIS software to project the cubes and generate files in the TIFF/GeoTIFF, which are compatible with MoonIndex.
- Filtration: The cubes are then filtered to reduce the noise. A Gaussian filter is applied in the spectral domain to enhance the absorption peaks of the main minerals. Then, a Fourier filter is applied in the frequency dimension to reduce the vertical striping common of M3 cubes.
- Continuum Removal: The tool allows the removal of the spectral continuum in a custom way. It can be done using a convex hull or a polynomial fit technique. Continuum removal is essential to subtract noise and background information from the spectral signature, highlighting the key absorption features essential for mineralogical analyses.
- Index Calculation: MoonIndex computes 38 spectral indexes recreated from the literature. Most of them operate over the bands to enhance the signal of a specific mineralogy, which is useful to do an initial survey of the surface composition. These indexes are organized into individual scripts for easier access, modification, and interpretation.
- Error Handling and Consistency Checks: Given M3 data artifacts, such as thermal and photometric inconsistencies, MoonIndex includes steps for adjusting and validating the data to ensure meaningful results across lunar terrains.
- Visualization: Results are exported as color composites (RGB maps) that can be viewed in QGIS or similar tools, allowing for efficient mineralogical mapping. The indexes can be exported as an individual RGB map, or as a single file containing all the indexes.

3.3. Workshops

To further prove the efficiency and utility of the Europlanet GMAP JupyterHub and the standalone docker images, we used them in several planetary data workshops. It was first tested during the "COSPAR capacity building workshop" in Antofagasta, Chile in 2023. In that occasion, the docker images were used locally to process Martian data sets, the test was done using several machines with different operating systems and specifications, proving the versatility of the GMAP JupyterHub.

The online platform was used during the "Planetary geology workshop" in La Paz, Bolivia in 2022, where 25 users stressed the server to test its capability. The platform performed reliably with up to 10 concurrent users. Performance suffered substantially as user numbers increased, causing a considerable rise in processing time. A slowdown ratio of approximately 1:n, where n denotes the number of active users, was observed. This indicates a proportional decrease in system responsiveness with each additional user. The underlying infrastructure for this test consisted of a single dual-socket server equipped with 40 cores (80 threads) at 2.2 GHz and 256 GB of ECC DDR4 RAM. We suggest deploying the infrastructure on a high-performance computing system (HPC) with a minimum of 512 GB RAM and 128 CPU cores and 256 threads (2.6–3.0 GHz) for optimal performance. Single-threaded processing steps especially benefit from faster clock speeds. For optimal efficiency and scalability, the ideal configuration uses Kubernetes to manage multiple HPC nodes, distributing workloads effectively. This is likely related to constraints in the memory and power of the server station, so a more powerful hosting facility would be able to allow more synchronous users. Finally, the platform was tested again, this time also with the MoonIndex tool, during the "First Latin American Planetary Science Workshop" in Buenos Aires, Argentina in 2023. In this occasion, six users worked with MoonIndex to generate spectral indexes from M3 cubes. This test worked whiteout problems, showcasing the educational and scientific potential of the platform.

Finally, we presented the platform and the ancillary docker images during the GMAP Winter School, in 2023 and 2024 (GMAP Team, 2023, 2024, 2025).

4. Discussions

4.1. Customization and Flexibility of User-Centric Environment

One of the standout features of the JupyterHub setup is its flexibility in environment customization. Through the activation of various Docker images, users are able to choose from a selection of environments customized for specific tasks or research requirements. For example, individuals have the ability to effortlessly transition between platforms dedicated to general data science, geospatial analysis, or planetary science resources. This flexibility negates the requirement for repetitive installations and configurations, ultimately preserving time and effort. Users can rapidly switch between diverse toolsets, thereby ensuring access to the required libraries,

software, and dependencies. This modular design facilitates the incorporation of updates and new tools, allowing administrators to seamlessly integrate new environments or update current ones without interrupting the overall deployment. Administrators have full control of customizations of the user's environment by implementing predefined settings, paths, and resources to align with their project needs. Consequently, the end user will benefit from an environment that is fully equipped and prepared, with all required dependencies and configurations preloaded, resulting in reduced time spent on the initial setup. For instance, default scripts can automatically load libraries, define data paths, or initialize specific plugins, thereby ensuring that each user session starts with the correct resources in place. This function not only aids in time management but also ensures consistency across sessions and users, streamlining result replication for researchers to emphasize analysis rather than technical preparation. By adopting a user-centric approach, the system becomes more accessible to users with different levels of technical expertise, thereby cultivating a more efficient and interactive research setting.

This platform proves especially beneficial in educational settings, including classrooms, workshops, and remote training, where instructors must demonstrate techniques or facilitate practical exercises for students. Centralizing the computational environment via JupyterHub removes the necessity for individual participants to install and configure intricate software, thereby mitigating a significant impediment to effective training. This approach is particularly advantageous in interdisciplinary domains; students in fields such as geology and planetary science can concentrate on data processing, visualization, and analysis, including remote sensing and spatial data—without extensive programming skills. Regardless of whether the objective is coding instruction, data science workflow training, or scientific data interpretation education, JupyterHub offers a user-friendly, web-based learning environment requiring only internet access and the necessary data sets.

4.2. Collaborative Science

Shared data spaces are critical for facilitating collaboration among scientists. In this deployment, storage is carefully divided into private and shared sections, enabling a blend of individual workspaces and collaborative assets. Shared data spaces are particularly valuable for team projects or research groups, where data sets, models, and analytical results need to be easily accessible to multiple users. Through centralized access to shared data, collaborators can simultaneously work on data sets, share insights, and contribute to ongoing analyses. By adopting this arrangement, various logistical impediments commonly encountered in traditional data sharing methods, such as file transfers or manual data synchronization, are eradicated, thereby ensuring widespread access to the most recent data iterations. Efficiency is improved and a culture of collaborative science is cultivated among interdisciplinary teams or geographically dispersed research groups through the shared data environment.

JupyterHub is Capable of hosting other functions and tools inside it, becoming an ecosystem where the user could find all the necessary steps to conduct their research. For example, a scientist interested in the geological mapping of a lunar location could process the images in ISIS, create DTMs in ASP, and generate spectral indexes in MoonIndex, obtaining enough information to overcome the task. This advantage becomes stronger as the JupyterHub is further scaled.

4.3. Scalability and Reproducibility

Ensuring scalability and reproducibility is crucial for handling and studying extensive data sets, particularly in disciplines such as planetary science where the amount of available data can be extremely large. The GMAP Docker-based JupyterHub setup is easily scalable due to its utilization of containerization, facilitating the seamless addition of resources or deployment across numerous servers if required. This system is capable of supporting an increasing number of users and a growing amount of data without compromising on performance. Efficiently scaling resources is vital for maintaining performance and managing data complexity in large-scale analyses, like processing planetary data sets from satellite or rover missions.

The use of Docker containers also contributes to improved reproducibility, as each Docker image serves as a snapshot of a fully configured environment. Preservation of the precise software versions, dependencies, and configurations allows for accurate replication of analyses by other users or at later dates. In research domains that prioritize reproducibility as a fundamental scientific concept, this ability is crucial, empowering scientists to authenticate results and share methodologies with certainty. Scholars have the capability to export or distribute their Docker images in official registry such as Docker Hub, as well as save the state of the container as a compressed file via docker commands. This allows other users to replicate the identical environment in any

Docker-compatible system, thus enhancing reproducibility across various organizations and computational frameworks.

Moreover, the platform's architecture is inherently compatible with contemporary orchestration tools, including Kubernetes and OpenShift, thus facilitating deployment across distributed multi-node systems. Enhanced scalability, achieved through dynamic resource allocation, load balancing, and fault tolerance, is a key benefit of this integration, as is the automated deployment of standardized environments at scale, which promotes reproducibility. Consequently, these features make the GMAP JupyterHub configuration well-suited to both small academic laboratories and large collaborative research initiatives.

The platform's use of standard Open Container Initiative (OCI)-compliant containers ensures broad system and infrastructure compatibility. Cross-platform compatibility enables deployment in any OCI-compliant runtime environment, such as Docker or Podman, thereby enhancing integration and portability. However, it should be noted that macOS and Windows platforms may introduce limitations related to native container execution, typically requiring intermediate layers such as virtual machines or Windows Subsystem for Linux (WSL2) for compatibility. However, no compatibility issues were identified for GMAP's specific images.

5. Conclusion and Future Work

The ongoing Docker-based JupyterHub deployment offers a stable groundwork for scientific research. Nonetheless, several enhancements and extensions are on the agenda to expand its capabilities, functionality, and durability. Future efforts will concentrate on broadening the range of Docker environments, incorporating new tools and libraries, and improving deployment strategies for long-lasting viability.

5.1. Enhancing Capabilities Through the Inclusion of More Docker Images

As the research environment evolves, it is imperative that the tools and platforms available to our users also evolve. Plans are in place to develop additional Docker images that address emerging needs in data science, geospatial analysis, and other specialized fields. These images may include:

- State-of-the-art Machine Learning and AI Frameworks: Containers incorporating TensorFlow, PyTorch, and scikit-learn to sustain deep learning, extensive model training, and advanced analytics.
- Environments optimized for High-Performance Computing (HPC) and GPU usage are tailored to harness GPU resources and HPC capabilities, facilitating researchers in efficiently tackling computationally intensive tasks like satellite image processing and large-scale simulations.
- Enhanced Geospatial and Planetary Science Tools: Improved toolkits designed for planetary science, GIS, and remote sensing, incorporating libraries and software that address new data sets and research inquiries.

5.2. Enhanced Data Management and Collaboration Features

To address the requirements of collaborative scientific initiatives, upcoming developments will enhance data management and sharing features. We aim to:

- Implement version-controlled data repositories which can allow researchers to monitor data modifications and rollback to earlier versions, facilitating reproducibility and streamlining collaborative processes.
- Incorporate Data Storage Extensions to expand storage options by connecting with external data warehouses, cloud storage solutions, and remote data sets, enabling researchers to access larger data sets and store results more flexibly.

5.3. Long-Term Deployment Strategy

We are actively researching advanced deployment architecture and infrastructure management strategies to ensure the sustained operational effectiveness of the JupyterHub deployment within scientific computing environments. These designs are implemented to ensure the continued scalability, maintainability, and efficiency of the system as user demand and data complexity increase.

- Dynamic management of Jupyter environments across distributed resources is facilitated by employing Kubernetes for container orchestration. Kubernetes provide auto-scaling, load balancing, and self-healing

capabilities for services, guaranteeing high availability and optimized resource utilization regardless of multi-user load or demand fluctuations.

- Continuous monitoring of CPU, memory, storage utilization, and container health will be enabled through the implementation of automated tools such as Prometheus for metrics collection and Grafana for data visualization. These systems offer capabilities for detecting performance bottlenecks, automating software updates, and ensuring robust backup and recovery processes, consequently lessening the operational workload and improving the overall system's reliability.
- Differential resource allocation, customized to user profiles (e.g., training, research, administrative) and project requirements, will optimize fairness and efficiency. This infrastructure's adaptability allows it to effectively support diverse functionalities, from preliminary data analysis to the execution of resource-intensive data processing tasks.

Furthermore, the achievement of consistent performance within Docker containers depends on several key factors.

- Standardized, pre-configured environments are implemented to ensure reproducibility, preventing dependency conflicts and version inconsistencies across users and projects.
- The performance of all users must be protected through resource isolation, mitigating the impact of high-computation tasks.
- Portability facilitates the consistent and efficient deployment of environments across multiple platforms (cloud, high-performance computing, and local clusters), resulting in predictable behavior.

In response to escalating demand, we project a transition to a federated model comprising interconnected HPC systems. These systems will be managed via Kubernetes or OpenShift, with each node contributing resources to a shared pool. This facilitates elastic scaling, fault tolerance, and inter-institutional collaboration, crucial for the long-term viability of data-intensive scientific workflows.

5.4. Community and Open-Source Collaboration

Future endeavors will involve cultivating a more robust community centered on the JupyterHub deployment through the promotion of feedback, teamwork, and open-source contributions. This will include:

- Introducing the release of new Docker images and extensions as open-source projects to encourage community engagement and participation.
- The implementation of user feedback channels and documentation enhancements involves the creation of channels within the Europlanet community and international planetary working groups. These channels allow users to propose features, report problems, and contribute to the code. Enhanced documentation will assist new users and facilitate the involvement of external contributors in the project.

Acknowledgments

Acknowledgment is given to the United States Geological Survey (USGS) for the provision of the Integrated Software for Imagers and Spectrometers (ISIS) (Sucharski et al., 2020), essential for data processing in this research. NASA's Ames Stereo Pipeline (ASP) (Beyer et al., 2018) is also acknowledged for its critical role in enabling efficient data analysis workflows. Appreciation is extended to the Planetary Data System Geoscience Node—Orbital Data Explorer (PDS-ODE) (PDS Geosciences Nodes, 2020) for maintaining comprehensive data archives of Mars and the Moon, which provided invaluable resources for this study. This study is within the Europlanet 2024 RI and EXPLORE projects, and it has received funding from the European Union's Horizon 2020 research and innovation programme under Grants 871149 and 101004214 and financial support from the ASI-INAF Agreement 2022-14-HH.0. Open Access funding enabled and organized by Projekt DEAL.

Conflict of Interest

The authors declare no conflicts of interest relevant to this study.

Data Availability Statement

The original data used for this study are available at publicly data archive NASA PDS Geoscience Node ODE (PDS Geosciences Nodes, 2020).

Software for this research is available on GitHub: https://github.com/europlanet-gmap/docker-jupyterhub and Zenodo: (Brandt et al., 2024).

Integrated Software for Imagers and Spectrometers (ISIS): (Laura et al., 2023) and NASA Ames Stereo Pipeline: (NASA, 2025).

References

Authentik Security Inc. (2025). GitHub - Goauthentik/Authentik: The authentication glue you need. Retrieved from https://github.com/goauthentik/authentik

Beg, M., Taka, J., Kluyver, T., Konovalov, A., Ragan-Kelley, M., Thiéry, N. M., & Fangohr, H. (2021). Using Jupyter for reproducible scientific workflows. *Computing in Science & Engineering*, *23*(2), 36–46. https://doi.org/10.1109/MCSE.2021.3052101

Beyer, R. A., Alexandrov, O., & McMichael, S. (2018). The AMES stereo pipeline: NASA's open source software for deriving and processing terrain data. *Earth and Space Science*, *5*(9), 537–548. https://doi.org/10.1029/2018EA000409

Bisong, E. (2019). Google colaboratory. In E. Bisong (Ed.), *Building machine learning and deep learning models on Google Cloud Platform: A comprehensive guide for beginners* (pp. 59–64). Apress.

Brandt, C. H., Nodjoumi, G., Pio Rossi, A., Luzzi, E., & Valencia, J. S. (2024). Europlanet-Gmap/Docker-JupyterHub: First release. Retrieved from https://zenodo.org/records/14555694

Brewer, E. A. (2015). Kubernetes and the path to cloud native. In *Proceedings of the sixth ACM symposium on cloud computing, SoCC '15* (p. 167). Association for Computing Machinery.

Cox, N., Poznanski, D., Eilat-Bloch, Y., Zijlstra, A., McDonald, I., Recio-Blanco, A., et al. (2022). EXPLORE-innovative scientific data exploration and exploitation applications for space sciences. In *SciOps 2022: Artificial Intelligence for science and operations in astronomy (SCIOPS). Proceedings of the ESA/ESO SCOPS workshop held 16-20 may* (Vol. 36).

Federal Geographic Data Committee. (1998). *Content standard for digital geospatial metadata (CSDGM)*. Federal Geographic Data Committee.

Forde, J., Head, T., Holdgraf, C., Panda, Y., Nalvarete, G., Ragan-Kelley, B., & Sundell, E. (2018). Reproducible research environments with Repo2Docker.

GMAP Team. (2023). Europlanet-Gmap/Winter-School-2023.

GMAP Team. (2024). Europlanet-Gmap/Winter-School-2024.

GMAP Team. (2025). *2025 winter school*. Europlanet GMAP Winter School. Retrieved from https://www.planetarymapping.eu/

Granger, B. E., & Pérez, F. (2021). Jupyter: Thinking and storytelling with code and data. *Computing in Science & Engineering*, *23*(2), 7–14. https://doi.org/10.1109/MCSE.2021.3059263

Heon, M., Walsh, D., Baude, B., Mohnani, U., Cui, A., Sweeney, T., et al. (2018). Podman - : A tool for managing OCI containers and pods.

Jupyter, P., Bussonnier, M., Forde, J., Freeman, J., Granger, B., Head, T., et al. (2018). *Binder 2.0 - Reproducible, interactive, sharable environments for science at scale*. Scipy. https://doi.org/10.25080/Majora-4af1f417-011

Karlitschek, F. (2018). *NextCloud*. Nextcloud. Retrieved from http://nextcloud.com

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., et al. (2016). Jupyter Notebooks–a publishing format for reproducible computational workflows. In *Positioning and power in academic publishing: Players, agents and agendas* (pp. 87–90). IOS press.

Kubernetes. (2025). GitHub - Kubernetes/Kubernetes: Production-grade container scheduling and management. Retrieved from https://github.com/kubernetes/kubernetes

Laura, J., Acosta, A., Addair, T., Adoram-Kershner, L., Alexander, J., Alexandrov, O., et al. (2023). Integrated software for imagers and spectrometers.

Lovell, L., Adriani, I. C., Nodjoumi, G., Suarez-Valencia, J. E., Le Corre, D., Heward, A., et al. (2024). Design of robotic traverses on the Archytas dome on the moon [version 1; peer review: 1 approved, 1 approved with reservations]. *Open Research Europe*, *4*(116), 116. https://doi.org/10.12688/openreseurope.17424.1

Luzzi, E., Heldmann, J. L., Williams, K., Deutsch, A., Sehlke, A., & Nodjoumi, G. (2023). Geomorphological evidence of near-surface ice at candidate landing sites in Arcadia Planitia, Mars. In *LPI contributions* (Vol. 2806, p. 2335).

Luzzi, E., Nodjoumi, G., Massironi, M., Pozzobon, R., & Rossi, A. P. (2023). A semi-automated analysis of displacement-to-length scaling of the Grabens affecting lunar floor-fractured craters. *Journal of Geophysical Research: Planets*, *128*(4), e2022JE007265. https://doi.org/10.1029/2022je007265

Merkel, D. (2014). Docker: Lightweight Linux containers for consistent development and deployment. *Linux Journal*, *2014*(239), 2. Retrieved from https://www.seltzer.com/margo/teaching/CS508.19/papers/merkel14.pdf

NASA. (2025). NeoGeographyToolkit/StereoPipeline: 2025-04-16-Daily-Build. Retrieved from https://zenodo.org/records/15230598

Nass, A., Massironi, M., Rossi, A., Penasa, L., Pozzobon, R., Brandt, C., et al. (2021). GMAP: European mapping efforts for geologic mapping of planetary bodies.

Nodjoumi, G., Pozzobon, R., Sauro, F., & Rossi, A. P. (2023). DeepLandforms: A deep learning computer vision toolset applied to a prime use case for mapping planetary skylights. *Earth and Space Science*, *10*(1), e2022EA002278. https://doi.org/10.1029/2022EA002278

Nodjoumi, G., Valencia, J. E., Le Corre, D., Lovell, L., Adriani, I. C., Heward, A., et al. (2023). Insights and outcomes of the 2022 explore machine learning lunar data challenge and next 2023 edition.

Nodjoumi, G., Valencia, J. E. S., Brandt, C., Cox, N. L. J., & Rossi, A. P. (2022). EXPLORE lunar scientific data applications: L-explo and L-Hex. In *European planetary science congress* (p. EPSC2022-966).

Open Geospatial Consortium. (2019). OGC API – Features – Part 1: Core. *Open Geospatial Consortium*.

PDS Geosciences Nodes. (2020). PDS geosciences node orbital data explorer (ODE). Retrieved from https://ode.rsl.wustl.edu/

Red Hat, Inc. (2025). OpenShift · GitHub. Retrieved from https://github.com/openshift

Suárez-Valencia, J. E., Rossi, A. P., Zambon, F., Carli, C., & Nodjoumi, G. (2024). MoonIndex, an open-source tool to generate spectral indexes for the moon from M3 data. *Earth and Space Science*, *11*(6), e2023EA003464. https://doi.org/10.1029/2023EA003464

Sucharski, T., Mapel, J., Lee, K., Shepherd, M., Combs, C. R., Stapleton, S., et al. (2020). USGS-Astrogeology/ISIS3: ISIS 4.2.0 public release.