SoftwareX 34 (2026) 102547

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

prismatools: An open-source Python package for accessing and analyzing PRISMA hyperspectral data

Lorenzo Crecco *, Sofia Bajocco

Council for Agricultural Research and Economics, Research Centre for Agriculture and Environment (CREA-AA), Italy

ARTICLE INFO

Keywords:
Satellite data
Earth observation
PRISMA
ASI
Python

ABSTRACT

Hyperspectral remote sensing captures hundreds of contiguous, narrow spectral bands in the VNIR and SWIR ranges, enabling detailed analysis of vegetation, water quality, soil properties, and other environmental variables. *prismatools* is an open-source Python package that facilitates processing, visualization, and analysis of PRISMA Level 2 products. It converts VNIR, SWIR and panchromatic PRISMA data into georeferenced xarray datasets, supporting seamless integration into workflows with other popular Python packages. The package also provides interactive mapping and spectral exploration leveraging the capabilities of the popular package Leafmap, along with methods for computing vegetation indices, performing PCA, extracting spectral signatures and exporting processed images.

Metadata

Nr	Code metadata description	*Metadata*
C1	Current code version	v0.1.4
C2	Permanent link to code/repository used for this code version	https://github.com/gthlor/prismatools
C3	Permanent link to reproducible capsule	-
C4	Legal code license	MIT License
C5	Code versioning system used	git
C6	Software code languages, tools and services used	Python
C7	Compilation requirements, operating environments and dependencies	python (≥3.10), affine, bqplot, h5py, ipyleaflet, leafmap, localtileserver, matplotlib, numpy, pandas, rioxarray, xarray, scikit-learn
C8	If available, link to developer documentation/manual	https://gthlor.github.io/prismatools
C9	Support email for questions	lorenzo.crecco@crea.gov.it

1. Motivation and significance

Hyperspectral remote sensing refers to the simultaneous acquisition of images in many narrow, contiguous spectral bands, providing detailed information across the visible (VIS), near-infrared (NIR), and shortwave infrared (SWIR) regions of the electromagnetic spectrum (typically, in the range of 400–2500 nm) [1]. Unlike multispectral sensors, which typically record fewer than ten broad bands (~100 nm bandwidth), hyperspectral sensors acquire hundreds of narrow bands (<20 nm). The resulting hyperspectral cube, composed of two spatial dimensions (x, y) and one spectral dimension (z), enables fine discrimination of surface properties based on their spectral signatures [2]. This capability has made hyperspectral remote sensing an essential tool in a wide range of applications, including agriculture and forestry [3,4], ecology [5], geology and mineral exploration [6], atmospheric and climate studies [7], cryosphere [8] and aquatic ecosystem monitoring [9].

Spaceborne hyperspectral systems typically have lower spatial resolution (30–150 m) than their airborne counterparts (35 cm–4 m) [10]. However, they provide broader spatial coverage, consistent revisit time, and cost-effective data acquisition over large regions, enabling continuous environmental monitoring at multiple spatial and temporal scales [2].

In recent years, several spaceborne hyperspectral missions have been launched, including the Italian PRecursore IperSpettrale della Missione Applicativa (PRISMA) [11], the German Environmental Mapping and Analysis Program (EnMAP) [12], and upcoming missions such as CHIME. Among these, PRISMA, launched by the Italian Space Agency (ASI) in 2019, provides global coverage with a spectral resolution of 12 nm across 239 contiguous bands (66 bands within 400–1010 nm in the VNIR and 173 bands within 920–2500 nm in the SWIR), a spatial

* Corresponding author.
E-mail address: lorenzo.crecco@crea.gov.it (L. Crecco).

https://doi.org/10.1016/j.softx.2026.102547
Received 10 November 2025; Received in revised form 30 December 2025; Accepted 27 January 2026
Available online 11 February 2026
2352-7110/© 2026 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC license (http://creativecommons.org/licenses/by-nc/4.0/).

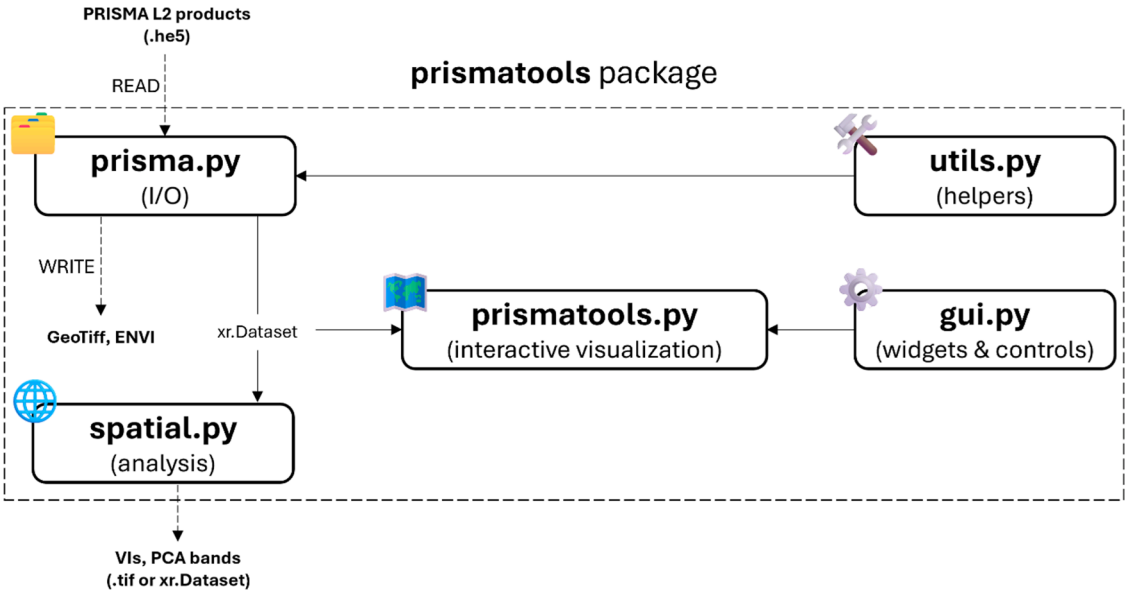


Fig. 1. Modular architecture of the *prismatools* Package. The diagram illustrates the functional organization of the *prismatools* package, showing the core modules and their respective tasks.

resolution of 30 m and a revisit time of 29 days, offering unprecedented opportunities for environmental monitoring. A panchromatic camera complements the system with a single band in the wavelength range of 400–700 nm at a 5 m spatial resolution [13].

Despite the potential of PRISMA data, effectively working with these products poses significant challenges, especially for non-expert users, due to the complex HDF5 data structure, the size of the datasets, and the lack of accessible tools for reading, processing, and visualizing hyperspectral information [13]. Currently, PRISMA HDF5 files can be opened with the EnMAP Toolbox [14], the ENVI-PRISMA Toolkit [13], and the PRISMA Toolbox [15]. However, these tools do not easily integrate into automated, open-source analysis pipelines, like, for instance, the modern Python geospatial ecosystem. More recently, the HyperCoast Python package [9] introduced a set of tools for interactive analysis of different hyperspectral datasets, including PRISMA, primarily aimed at supporting coastal ecosystem studies. Nonetheless, HyperCoast is limited to working with Level 2D (geolocated and geocoded surface reflectance) products and does not support Level 2B (geolocated on-ground radiance) or Level 2C (geolocated reflectance) products. Moreover, it does not include functions for extracting panchromatic data, useful for pan-sharpening VNIR and SWIR bands, or exporting outputs beyond GeoTIFF format. Finally, the prismaread R library provides basic PRISMA data reading and interaction functionalities but is currently not actively maintained [16].

To address these challenges, we developed *prismatools*, a lightweight and open-source Python package designed to simplify access and processing of PRISMA Level 2 hyperspectral and panchromatic data. The software integrates efficient routines for reading and writing PRISMA products, interactive visualization using Leaflet-based maps, and spatial analysis tools such as spectral index computation and principal component analysis (PCA), thereby making hyperspectral data analysis more accessible to the broad scientific community.

2. Software description

prismatools is implemented in Python ≥3.10. It is platform-independent (noarch), and builds on widely used Python packages, including xarray, rioxarray, numpy, h5py, affine, ipyleaflet, scikit-learn, bqplot, localtileserver and leafmap. If Jupyter is installed in a separate environment, ipyleaflet must be installed in the Jupyter environment and bqplot must be enabled as a Jupyter extension to ensure proper rendering of interactive maps.

The package is available on PyPI and can be installed easily using:

```
1. pip install prismatools
```

which provides the latest stable release. It is also available on conda-forge, and can be installed either directly using:

```
1. conda install -c conda-forge prismatools
```

or within a dedicated environment to avoid conflicts with other packages:

```
1. conda create -n prisma python=3.10
2. conda activate prisma
3. conda install -c conda-forge prismatools
```

For developers, the latest development version can be installed directly from GitHub using:

```
1. pip install git+https://github.com/gthlor/prismatools
```

2.1. Software architecture

prismatools is organized into five components (or modules), as illustrated in Fig. 1: *prisma.py* which handles data input and output operations, including the reading of PRISMA Level-2 products; *prismatools.py* which manages interactive visualization as well as spectral exploration and extraction; *gui.py* which provides widgets and controls for user interaction; *spatial.py* which implements spatial analysis functionalities and *utils.py* which contains helper functions that support the other modules (mainly *prisma.py*).

2.2. Software functionalities

2.2.1. prisma.py module

The *prisma.py* module provides a set of functions to read, process, extract and export hyperspectral and panchromatic data from PRISMA Level 2 products (L2B, L2C and L2D). The functions read_prismaL2D and read_prismaL2BC read the original HDF5 files, allowing users to choose between hyperspectral (VNIR+SWIR) and panchromatic cubes. These functions apply the appropriate scaling from digital numbers to surface reflectance, handle fill values, and build an xarray.Dataset with dimensions x, y and wavelength (hyperspectral) or x and y

(panchromatic). They also allow the optional selection of specific wavelengths through exact or nearest-neighbor matching (Supplementary Listing 1) and the import of the PRISMA Level 2 error matrix as a dedicated data variable within the resulting xarray.Dataset, enabling pixel-level quality masking and filtering.

The functions write_prismaL2D and write_prismaL2BC convert either a PRISMA dataset (as an xarray.Dataset) or a .he5 file path into georeferenced raster images, supporting the GeoTIFF, Cloud Optimized GeoTIFF, and ENVI formats. Similar to the reading functions, they handle both single-band panchromatic and hyperspectral cubes and, for the latter, allow the optional selection of wavelengths before writing (Supplementary Listing 2). Output raster values are stored as floating-point surface reflectance, and the nodata value is set to 0.

The helper functions array_to_image and extract_prisma handle the actual conversion of a Numpy array (2D o 3D) into a georeferenced raster file (automatically applying the appropriate data type and writing band metadata) and the extraction of average reflectance values from a given region of interest, respectively (Supplementary Listing 3).

2.2.2. *prismatools.py module*

The *prismatools.py* module extends the *leafmap* package [17] to provide an interactive interface for visualizing and analyzing PRISMA data within a Jupyter or Google Colab environment. It defines a custom Map class that inherits from leafmap.Map and integrates the reading and writing routines defined in *prisma.py* with *ipyleaflet* and *leafmap* widgets, allowing users to load, display, and explore PRISMA data directly on an interactive map. Through the add_prisma method, the module reads either a PRISMA HDF5 file or an in-memory xarray.Dataset, optionally selects specific wavelengths, converts the selected data to a temporary georeferenced raster, and renders it as a map layer. The integration with *ipyleaflet* enables interactive point sampling and real-time spectral visualization through dedicated widgets, offering a lightweight and user-friendly environment for the exploration of PRISMA hyperspectral cubes.

2.2.3. *gui.py module*

The *gui.py* module implements the graphical user interface (GUI) used in the *prismatools.py* module, providing a set of interactive widgets and controls for visualization, layer management, and spectral analysis. The main_toolbar function creates the primary control bar on the map, managing tools for basemap selection and layer visualization, while the layer_manager_gui function offers dynamic layer control with opacity adjustment and visibility toggling.

The spectralWidget class integrates the extract_prisma routine (*prisma.py*) to enable point-based spectral extraction: by clicking on the map, users can visualize reflectance spectra from PRISMA L2D data in real time through an interactive bqplot chart. The widget also supports multiple point comparisons, plot resetting, and spectral export in CSV format. Part of the code of the spectralWidget is adapted from the HyperCoast package [9].

2.2.4. *utilis.py module*

The *utils.py* module provides helper functions that support the core routines of the *prismatools* package by handling validation, coordinate transformations, and spatial metadata generation. Specifically, the check_valid_file function verifies the integrity and naming consistency of PRISMA Level 2 products before processing, convert_coords performs coordinate transformations between geographic and projected reference systems, while the get_transform function generates affine transformations based on upper-left corner coordinates and spatial resolution, enabling correct georeferencing of rasterized outputs.

2.2.5. *spatial.py module*

The *spatial.py* module implements spatial analysis and transformation functions for PRISMA hyperspectral data, providing tools for both spectral index computation and dimensionality reduction. The band_math function enables the derivation of custom spectral indices from hyperspectral datasets using user-defined algebraic expressions and wavelength mappings, allowing flexible computation of vegetation and biophysical indicators directly from Level 2 reflectance data. The pca_image function performs Principal Component Analysis (PCA) on hyperspectral (or multispectral images) to extract dominant spectral features while preserving spatial metadata. It supports both xarray.Dataset and GeoTIFF inputs, generating georeferenced principal component maps and explained variance statistics.

3. Illustrative examples

To illustrate the application of *prismatools*, we describe two examples: the interactive visualization and the spatial analysis of PRISMA L2D data. PRISMA datasets can be downloaded from the ASI PRISMA portal (https://prisma.asi.it/) after registering.

3.1. *Spectral index computation and PCA*

Surface objects have characteristic spectral signatures depending on their physical properties and interactions with electromagnetic radiation. Since environmental factors can modify these interactions, spectral signatures can provide insights into surface processes such as vegetation dynamics [18], urban expansion [19], or fire disturbance [20]. Spectral indices are mathematical combinations of reflectance values in specific spectral bands designed to highlight surface properties or processes by reducing confounding effects and isolate specific signals [21]. For example, vegetation indices such as the Normalized Difference Vegetation Index (NDVI; [22]), the Soil Adjusted Vegetation Index (SAVI; [23]) and the Enhanced Vegetation Index (EVI; [24]) are commonly used to assess vegetation vigor, chlorophyll content, and canopy structure. Similarly, the Normalized Difference Moisture Index (NDMI; [25]) and the Normalized Burn Ratio (NBR; [26]) have been designed to detect canopy water stress and burned areas, respectively.

Once the user has downloaded a PRISMA L2D dataset (without changing the first 11 characters of the file name which will be used for checking purposes), the user can compute indices (e.g. NDVI) as follows:

```
from prismatools import read_prismaL2D
from prismatools.spatial import band_math, pca_image
file = "/path/to/prisma.he5"  # example: r"..\..\data\PRS_L2D_STD_20240429095823_0001.he5"

# read PRISMA L2D dataset
ds = read_prismaL2D(file_path=file)

# define bands, constants and the index
wavelength_dict = {'red': 660, 'nir': 840}
expressions =  {'NDVI' :'(nir - red) / (nir + red)'}
constants = {}

# compute the index
result = band_math(ds, expressions, wavelength_dict, constants, inplace=True)
print(result)
result.NDVI.plot(cmap='YlGn')

```

Optionally, the user can also compute multiple indices (e.g. SAVI, EVI, NDVI) at once and export them (Supplementary Listing 4).

PCA can be performed from a GeoTIFF file or an xarray.Dataset as illustrated:

```
input_file = "path/to/TIFF_image.tif"  # replace with actual path
output_file = "path/to/out_TIFF_image.tif"  # replace with actual path

# from GeoTIFF (file path)
n_components = 10
ds_pca, expl_var = pca_image(input_file, output_file, n_components=n_components)
print(ds_pca)
print(expl_var)

# from xarray.Dataset
n_components = 10
ds_pca, expl_var = pca_image(ds, output_file, n_components=n_components)
print(ds_pca)
print(expl_var)

```

3.2. Interactive visualization of PRISMA L2D data

After PRISMA L2D dataset is read as an xarray.Dataset and the interactive map is created as follows:

```
import prismatools

file = "/path/to/prisma.he5"  # example: r"..\..\data\PRS_L2D_STD_20240429095823_0001.he5"

# read PRISMA L2D dataset
ds = prismatools.read_prismaL2D(file_path=file)

# create an interactive map
m = prismatools.Map()
m
```

The user can add the hyperspectral cube and load the spectral widgets to interact with it (e.g. plotting the spectral signatures as shown in Fig. 2).

```
1. m.add_prisma(ds, wavelengths=[650.0, 550.0, 450.0], vmin=0, vmax=0.2)
2. m.add("spectral")
3. m
4.
```

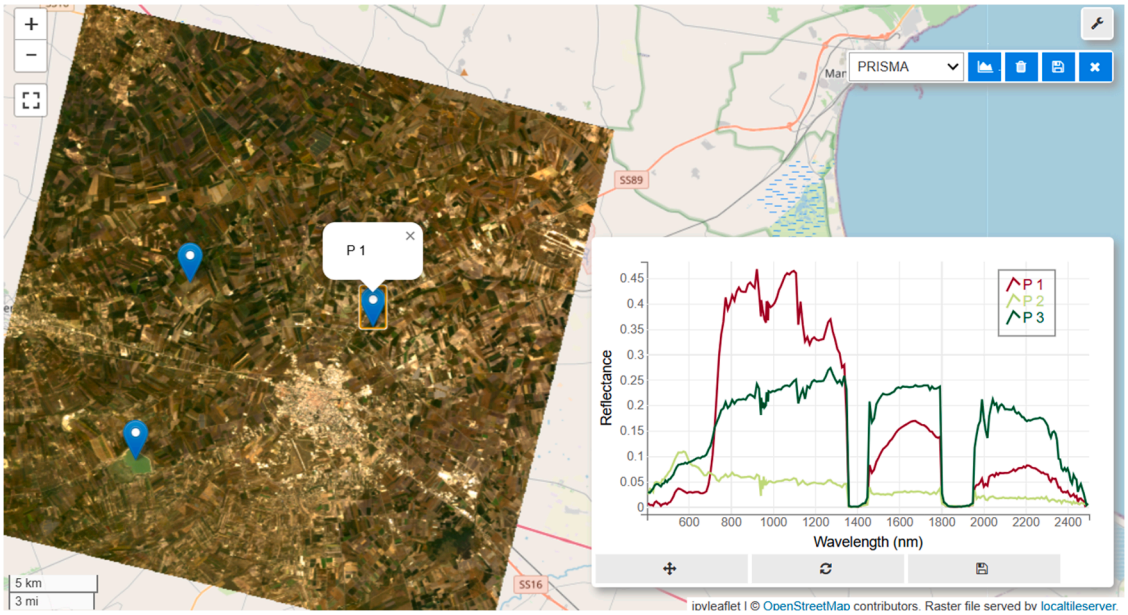


Fig. 2. The PRISMA hyperspectral cube is displayed on an interactive Leaflet map, illustrating the visualization of spectral signatures of sampled points.

4. Impact

Hyperspectral remote sensing provides unparalleled spectral detail but poses challenges in data accessibility, visualization, and analysis, particularly for complex formats as those provided by the PRISMA mission [13]. Existing solutions rely mainly on proprietary or mission-specific software (e.g., ENVI-PRISMA toolkit, EnMAP Toolbox), which offer powerful processing capabilities but are not easily integrated into open and automated geospatial workflows.

prismatools addresses these limitations by offering a lightweight (see Supplementary Material S2), open-source Python package that simplifies the access, visualization, and analysis of PRISMA L2 hyperspectral and panchromatic data.

Key features include:

- Read and Write PRISMA hyperspectral and panchromatic L2 products (L2D, L2B, L2C);
- Interactive mapping and real-time spectral visualization;
- Export of processed data to standard geospatial formats (GeoTIFF, ENVI format);
- Analyze hyperspectral imagery (band math, vegetation indices, PCA);
- Full compatibility with popular Python libraries (xarray, rioxarray, leafmap).

By lowering technical barriers, *prismatools* facilitates the integration of PRISMA data into automated, reproducible geospatial workflows. It enables researchers to perform interactive hyperspectral analysis without proprietary tools and supports new research directions in environmental monitoring, agriculture and forestry, geology, and water quality assessment. The package, publicly available on GitHub, PyPI and conda-forge, promotes open science and encourages community-driven extensions for future hyperspectral missions, such as CHIME, or additional spatial analysis.

5. Conclusions

In this study, we presented *prismatools*, an open-source Python package designed to streamline the handling and analysis of PRISMA hyperspectral data. The package provides efficient routines for reading and writing HDF5 products, visualizing hyperspectral cubes interactively, and performing common analytical tasks such as spectral index computation and PCA.

By offering a lightweight and fully open alternative to proprietary solutions, *prismatools* lowers the entry barrier for hyperspectral data processing and promotes reproducible research practices. Future developments will focus on expanding support to upcoming hyperspectral missions (e.g., CHIME) and integrating additional analysis algorithms contributed by the research community.

CRediT authorship contribution statement

Lorenzo Crecco: Writing – original draft, Software, Methodology, Conceptualization. **Sofia Bajocco:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The work was carried out within the Agritech National Research Center and received funding from the European Union Next-Generation EU (National Recovery and Resilience Plan (NRRP) – MISSION 4 COMPONENT 2, INVESTMENT 1.4 – D.D. 1032 17/06/2022, CN00000022). The work was also carried out within the AGRARSENSE (Smart, digitalized components and system for data-based Agriculture and Forestry – P101095835) project funded by the European Union. This manuscript reflects only the author's views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.softx.2026.102547.

References

[1] Goetz AFH, Vane G, Solomon JE, Rock BN. Imaging spectrometry for earth remote sensing. Science 1985;228:1147–53. https://doi.org/10.1126/science.228.4704.1147.
[2] Govender M, Chetty K, Bulcock H. A review of hyperspectral remote sensing and its application in vegetation and water resource studies. WSA 2009:33. https://doi.org/10.4314/wsa.v33i2.49049.
[3] Lu B, Dao P, Liu J, He Y, Shang J. Recent advances of hyperspectral imaging technology and applications in agriculture. Remote Sens 2020;12:2659. https://doi.org/10.3390/rs12162659.
[4] Adão T, Hruška J, Pádua L, Bessa J, Peres E, Morais R, Sousa J. Hyperspectral imaging: a review on UAV-based sensors, data processing and applications for agriculture and forestry. Remote Sens 2017;9:1110. https://doi.org/10.3390/rs9111110.
[5] Wang K, Franklin SE, Guo X, Cattet M. Remote Sensing of ecology, biodiversity and conservation: a review from the perspective of Remote sensing specialists. Sensors 2010;10:9647–67. https://doi.org/10.3390/s101109647.
[6] Yu H, Kong B, Wang Q, Liu X, Liu X. Hyperspectral remote sensing applications in soil: a review. Hyperspectral Remote Sensing. Elsevier; 2020. p. 269–91. ISBN 978-0-08-102894-0.
[7] Calin MA, Calin AC, Nicolae DN. Application of airborne and spaceborne hyperspectral imaging techniques for atmospheric research: past, present, and future. Appl Spectrosc Rev 2021;56:289–323. https://doi.org/10.1080/05704928.2020.1774381.
[8] Salvatori R, Salzano R, Valt M, Cerrato R, Ghergo S. The collection of hyperspectral measurements on snow and ice covers in polar regions (SISpec 2.0). Remote Sens 2022;14:2213. https://doi.org/10.3390/rs14092213.
[9] Liu B, Wu Q. HyperCoast: a Python package for visualizing andAnalyzing hyperspectral data in Coastal environments. JOSS 2024;9:7025. https://doi.org/10.21105/joss.07025.
[10] Thenkabail PS, Lyon JG, editors. Hyperspectral Remote Sensing Of Vegetation. 0 ed. CRC Press; 2016. ISBN 978-1-4398-4538-7.
[11] Pignatti S, Acito N, Amato U, Casa R, Castaldi F, Coluzzi R, De Bonis R, Diani M, Imbrenda V, Laneve G, et al. Environmental products overview of the Italian hyperspectral Prisma mission: the SAP4PRISMA Project. In: Proceedings of the 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS). IEEE; July 2015. p. 3997–4000.
[12] Guanter L, Kaufmann H, Segl K, Foerster S, Rogass C, Chabrillat S, Kuester T, Hollstein A, Rossner G, Chlebek C, et al. The EnMAP Spaceborne Imaging spectroscopy Mission for Earth observation. Remote Sens 2015;7:8830–57. https://doi.org/10.3390/rs70708830.
[13] Shaik RU, Periasamy S, Zeng W. Potential assessment of PRISMA hyperspectral imagery for remote sensing applications. Remote Sens 2023;15:1378. https://doi.org/10.3390/rs15051378.
[14] Van Der Linden S, Rabe A, Held M, Jakimow B, Leitão P, Okujeni A, Schwieder M, Suess S, Hostert P. The EnMAP-Box—a toolbox and application programming interface for EnMAP data processing. Remote Sens 2015;7:11249–66. https://doi.org/10.3390/rs70911249.
[15] Agrimano L, Santoro F, Amoruso L, Abbattista C, Drimaco D. PRISMA toolbox, a tool for ingestion, interaction and processing of hyperspectral EO data. In: Proceedings of the IAF Earth Observation Symposium. International Astronautical Federation (IAF); 2024. p. 885–8.
[16] Busetto, L.; Ranghetti, L. Prismaread: a tool for facilitating access and analysis of PRISMA L1/L2 Hyperspectral Imagery v1.0.0 2020.
[17] Wu Q. Leafmap: a Python package for interactive mapping and geospatial analysis with minimal coding in a jupyter environment. JOSS 2021;6:3414. https://doi.org/10.21105/joss.03414.
[18] Tucker CJ. Red and photographic infrared linear combinations for monitoring vegetation. Remote Sens Env 1979;8:127–50. https://doi.org/10.1016/0034-4257(79)90013-0.
[19] Zhao Y, Zhu Z. ASI: an artificial surface index for landsat 8 imagery. Int J Appl Earth Obs Geoinf 2022;107:102703. https://doi.org/10.1016/j.jag.2022.102703.
[20] Smith AMS, Wooster MJ, Drake NA, Dipotso FM, Falkowski MJ, Hudak AT. Testing the potential of multi-spectral remote sensing for retrospectively estimating fire severity in African savannahs. Remote Sens Env 2005;97:92–115. https://doi.org/10.1016/j.rse.2005.04.014.
[21] Montero D, Aybar C, Mahecha MD, Martinuzzi F, Söchting M, Wieneke S. A standardized catalogue of spectral indices to advance the use of remote sensing in Earth system research. Sci Data 2023;10:197. https://doi.org/10.1038/s41597-023-02096-0.
[22] Rouse JW, Haas RH, Schell JA, Deering D. Monitoring vegetation systems in the great plains with ERTS. In: Proceedings of the NASA Special publication; 1974.
[23] Huete AR. A soil-adjusted vegetation index (SAVI). Remote Sens Env 1988;25:295–309. https://doi.org/10.1016/0034-4257(88)90106-X.
[24] Huete A, Didan K, Miura T, Rodriguez EP, Gao X, Ferreira LG. Overview of the radiometric and biophysical performance of the MODIS vegetation indices. Remote Sens Env 2002;83:195–213. https://doi.org/10.1016/S0034-4257(02)00096-2.
[25] Wilson EH, Sader SA. Detection of forest harvest type using multiple dates of landsat TM imagery. Remote Sens Env 2002;80:385–96. https://doi.org/10.1016/S0034-4257(01)00318-2.
[26] Benson NC, Key CH. Landscape Assessment: Ground Measure Of Severity, The Composite Burn Index, And Remote Sensing Of Severity, The Normalized Burn Ratio. 2006. USGS General Technical Report RMRS-GTR-164-CD.