The Journal of Open Source Software

Ndvi2Gif: A Python Package for Multi-Seasonal Remote Sensing Analysis via Google Earth Engine

Diego García Díaz [1]

1 Estación Biológica de Doñana (EBD-CSIC), Spain

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 29 December 2025
Published: unpublished

Summary

Google Earth Engine (GEE) is a cloud-based platform for planetary-scale geospatial analysis (Gorelick et al., 2017). Ndvi2Gif is a Python package built on top of GEE and geemap (Wu, 2020) that simplifies the generation and analysis of multi-temporal composite images from satellite and climate reanalysis data. The package provides unified access to 7 satellite platforms (Sentinel-1/2/3, Landsat 4–9, MODIS) and climate reanalysis data (ERA5-Land, CHIRPS), with 52 predefined spectral and radar indices plus 48 climate variables.

The core concept of Ndvi2Gif is the creation of **multi-seasonal statistical composites**. The `get_year_composite()` method returns an `ee.ImageCollection` where each image represents one year, with bands for each temporal period (e.g., `periods=12` generates bands named `january`, `february`, …, `december`). This band-based temporal organization enables queries such as: *What was the maximum flood extent detected in January across 40 years of Landsat (1984–2024)?* by simply calling `collection.select('january').max()`. Built-in zonal statistics extraction via `get_stats()` allows direct computation of statistics per polygon without external GIS processing.

Statement of need

Generating multi-temporal composites in GEE requires extensive code for collection filtering, cloud masking, scaling to reflectance, and temporal aggregation. The following example shows the standard approach to create seasonal NDVI composites from Landsat:

```
import ee
ee.Initialize()

roi = ee.Geometry.Point([-5.9, 37.2]).buffer(10000)

# Scale Landsat 8-9 (OLI) to surface reflectance
def scale_OLI(image):
    optical = image.select(['SR_B2','SR_B3','SR_B4','SR_B5','SR_B6','SR_B7']) \
                   .multiply(0.0000275).add(-0.2) \
                   .rename(['Blue','Green','Red','Nir','Swir1','Swir2'])
    return image.addBands(optical, None, True)

# Scale Landsat 4-7 (TM/ETM+) to surface reflectance
def scale_ETM(image):
    optical = image.select(['SR_B1','SR_B2','SR_B3','SR_B4','SR_B5','SR_B7']) \
                   .multiply(0.0000275).add(-0.2) \
                   .rename(['Blue','Green','Red','Nir','Swir1','Swir2'])
    return image.addBands(optical, None, True)
```

```
# Cloud mask using QA_PIXEL band
def maskClouds(image):
    qa = image.select('QA_PIXEL')
    mask = qa.bitwiseAnd(1 << 3).eq(0).And(qa.bitwiseAnd(1 << 4).eq(0))
    return image.updateMask(mask)

def addNDVI(image):
    return image.addBands(
        image.normalizedDifference(['Nir', 'Red']).rename('NDVI'))

def getSeasonalComposite(startDate, endDate):
    l8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2') \
           .filterBounds(roi).filterDate(startDate, endDate) \
           .map(maskClouds).map(scale_OLI).map(addNDVI)
    l5 = ee.ImageCollection('LANDSAT/LT05/C02/T1_L2') \
           .filterBounds(roi).filterDate(startDate, endDate) \
           .map(maskClouds).map(scale_ETM).map(addNDVI)
    return l8.merge(l5).select('NDVI').max()

composite = ee.Image.cat([
    getSeasonalComposite('2013-01-01', '2013-03-31'),
    getSeasonalComposite('2013-04-01', '2013-06-30'),
    getSeasonalComposite('2013-07-01', '2013-09-30'),
    getSeasonalComposite('2013-10-01', '2013-12-31')
])
```

The 40 lines above can be reduced to 5 lines using Ndvi2Gif:

```
from ndvi2gif import NdviSeasonality

ndvi = NdviSeasonality(roi=roi, start_year=2013, end_year=2013,
                       sat='Landsat', index='ndvi', periods=4)
collection = ndvi.get_year_composite()
```

This simplification extends across 7 satellite platforms, 52 spectral and radar indices, 48 climate variables, multiple temporal aggregation methods, and flexible region of interest specifications—all through a unified API that inherits geemap's interactive mapping capabilities. The architectural decisions and design philosophy that enable this simplification are detailed in the sections below. Figure 1 provides an overview of the package's workflow and capabilities.

The Journal of Open Source Software

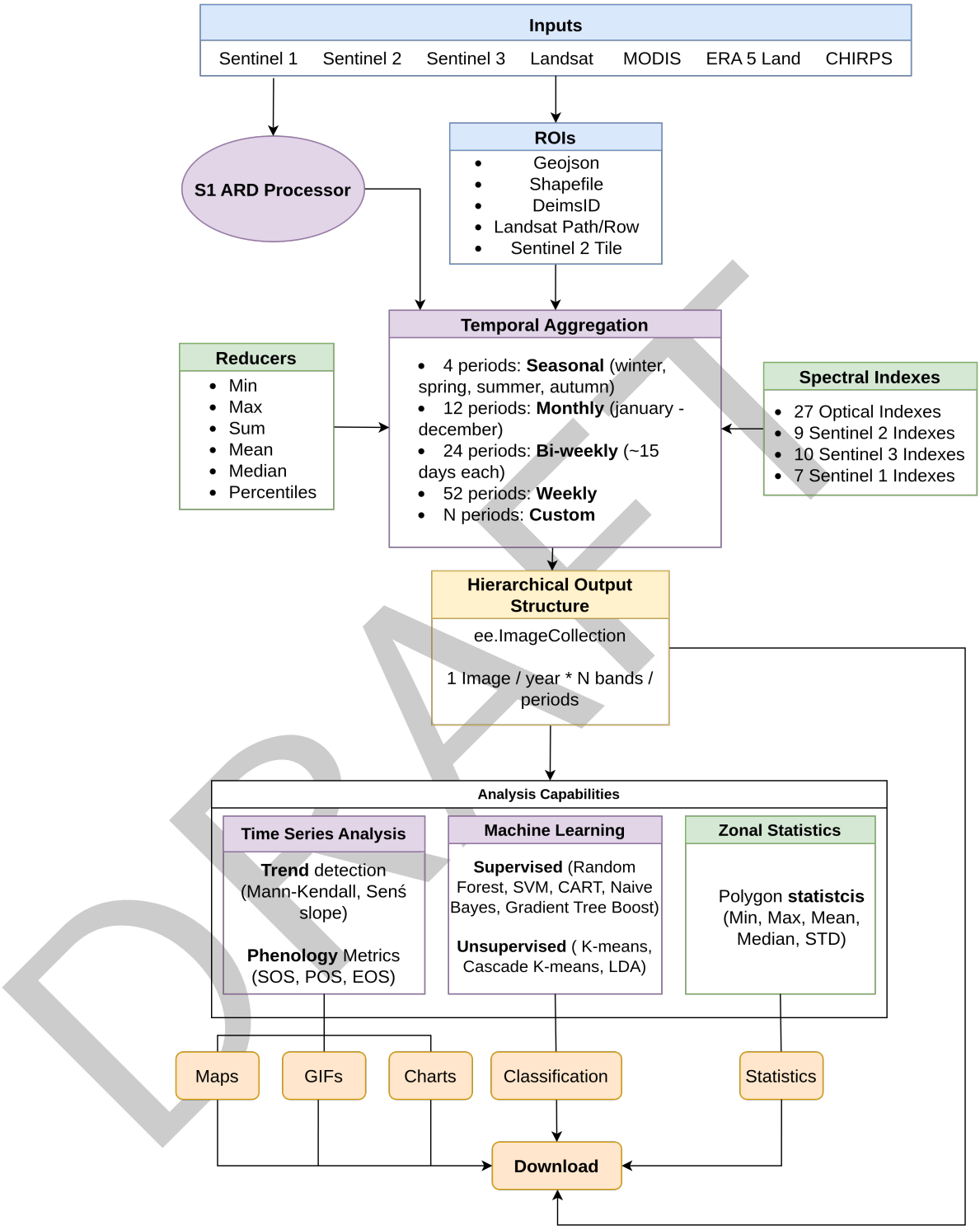

Figure 1: Ndvi2Gif workflow architecture and capabilities. Color coding: Blue — input data sources and ROI specification; Green — processing operations; Purple — main analysis classes; Orange — download-ready output products.

State of the Field

Remote sensing analysis using Google Earth Engine (GEE) has been widely facilitated by several Python-based libraries. Packages such as **geemap** provide interactive mapping and user-friendly access to Earth Engine data, while **eemont** focuses on preprocessing utilities and spectral index computation while preserving the native `ee.ImageCollection` temporal structure. **Wxee** emphasizes interoperability with the scientific Python ecosystem by exporting Earth Engine data into xarray objects for sequential time series analysis.

While these tools significantly lower the entry barrier to GEE, they are primarily designed as general-purpose interfaces or extensible utility layers. As a result, researchers conducting long-term ecological or phenological studies often need to implement substantial custom code to ensure consistent temporal alignment, multi-seasonal comparability, and reproducibility across decades of observations.

Ndvi2Gif addresses this gap by adopting a distinct design philosophy centered on **band-based temporal organization**, where each image represents a temporal unit (typically a year) and bands encode fixed intra-annual periods (e.g., months or seasons). This approach enables direct multi-decadal queries across consistent seasonal windows, which is difficult to achieve efficiently using the native collection-based paradigm.

Contributing this functionality to existing packages would require fundamental changes to their core temporal abstractions and design goals. Ndvi2Gif therefore complements, rather than replaces, existing GEE Python tools by providing a specialized framework tailored to multi-seasonal ecological analysis, phenology validation, and operational environmental monitoring.

Key features

Multi-platform data access

Ndvi2Gif provides unified access to optical, SAR, and climate reanalysis datasets through a consistent interface (Table 1).

Table 1: Supported satellite and reanalysis datasets.

Platform	Collection	Resolution	Coverage	Variables
Sentinel-2	S2 SR HARMONIZED	10/20m	2015–present	40+ optical
Sentinel-3	OLCI	300m	2016–present	10 water quality
Landsat	L4–L9 Col. 2	30m	1982–present	40+ optical
MODIS	Terra/Aqua	500m	2000–present	40+ optical
Sentinel-1	GRD ARD	10m	2014–present	7 SAR
ERA5-Land	Hourly/Monthly	11km	1950–present	47 climate
CHIRPS	Daily	5.5km	1981–present	Precipitation

Flexible region of interest

The library supports multiple ROI specification formats, enabling seamless integration with existing workflows and the eLTER research network.

Table 2: Supported region of interest formats.

Format	Example	Description
Shapefile	`'study_area.shp'`	Local vector file

Format	Example	Description
GeoJSON	`'boundaries.geojson'`	GeoJSON file
DEIMS ID	`'deimsid:8eda49e9...'`	eLTER site (Wohner et al., 2019)
S2 tile	`'s2:29SQB'`	MGRS tile code
Landsat WRS	`'wrs:200,32'`	Path and Row
ee.Geometry	`ee.Geometry.Point()`	GEE geometry
geemap	`Map.draw_features`	Interactive selection

Advanced SAR processing

The `S1ARDProcessor` module implements a complete Analysis Ready Data (ARD) pipeline for Sentinel-1 imagery, based on Vollrath et al. (2020). This addresses a critical limitation of the default Sentinel-1 products available in Google Earth Engine, which do not include radiometric terrain correction or advanced speckle filtering.

The ARD workflow includes:

- **Radiometric terrain correction** using an angular-based approach with configurable digital elevation models (Copernicus and SRTM), essential for reducing topographic effects in mountainous or heterogeneous terrain.
- **Speckle filtering** using multiple algorithms (Refined Lee, Lee, Gamma MAP, Lee Sigma, Boxcar), allowing users to balance noise reduction and edge preservation depending on the application.
- **Scattering model selection**, supporting both volume scattering (vegetation-dominated surfaces) and surface scattering (bare soil or water).

This preprocessing enables the reliable use of SAR-derived indices for vegetation structure monitoring, surface moisture assessment, and land cover analysis in areas where optical data are limited by cloud cover.

Time series and phenology analysis

The `TimeSeriesAnalyzer` module provides a comprehensive framework for extracting, analysing, and visualising temporal dynamics from the multi-seasonal composite stacks generated by `NdviSeasonality`. Unlike pixel-level time series derived directly from raw `ee.ImageCollection` objects that encode time implicitly as a sequence of images, this approach operates on temporally aligned composite periods stored explicitly as bands within each yearly image, ensuring consistency across years and reducing sensitivity to irregular acquisition frequency or missing observations.

The module supports:

- **Trend detection and significance testing**, including non-parametric Mann–Kendall tests, Sen's slope estimation, and linear regression with confidence intervals.
- **Phenological metrics extraction**, such as Start of Season (SOS), End of Season (EOS), Peak of Season (POS), season length, amplitude, and growth and senescence rates, derived from smoothed seasonal trajectories.
- **Seasonal and interannual comparison**, enabling direct assessment of variability and long-term change across fixed temporal windows.
- **Integrated visual diagnostics**, producing multi-panel dashboards that combine time series plots, seasonal distributions, trend summaries, autocorrelation, and data quality indicators.

This design allows researchers to move seamlessly from data extraction to exploratory and quantitative temporal analysis within a single, reproducible workflow, and is particularly suited to long-term ecological monitoring, phenology validation, and climate–vegetation interaction studies.

Machine learning classification

The `LandCoverClassifier` module provides supervised and unsupervised land cover classification workflows based on multi-temporal composite stacks generated by `NdviSeasonality`. Supported supervised algorithms include Random Forest, Support Vector Machines (SVM), CART, and Gradient Tree Boost, while unsupervised approaches include K-means and Cascade K-means clustering.

The module supports feature stack generation from multi-seasonal indices, optional normalization, training and validation dataset handling, and accuracy assessment through confusion matrices and standard performance metrics. Feature importance analysis is also available for supervised classifiers, enabling interpretation of which temporal periods or indices contribute most strongly to classification results.

These capabilities allow Ndvi2Gif to function not only as a data extraction tool, but also as an integrated framework for exploratory and applied land cover analysis within the Google Earth Engine ecosystem.

Software Design

Several Python packages extend GEE for remote sensing workflows, including eemont (Montero, 2021) and wxee (Zuspan, 2021). While these tools can be combined to perform similar tasks, Ndvi2Gif (first released in May 2020) adopts a fundamentally different temporal organization strategy.

Ndvi2Gif preserves the native `ee.ImageCollection` abstraction used by Google Earth Engine, but reinterprets its internal temporal structure. Rather than representing time as a long sequence of individual images (e.g., one image per month), ndvi2gif implements a **hierarchical band-based approach** in which each element of the `ee.ImageCollection` corresponds to a temporal unit (typically a year), and bands within each image encode fixed intra-annual sub-periods (e.g., months, biweekly intervals, or seasons). For example, a 40-year monthly analysis produces 40 images with 12 bands each, rather than 480 individual monthly images.

This design enables queries such as: *What is the mean cyanobacteria index (NDCI) detected from Sentinel-2 during August across the last 10 years?* by simply calling `collection.select('august').mean()`, or *In which week of the year does chlorophyll detected from Sentinel-3 peak?* by computing `collection.max().bandNames()`.

This architecture trades increased per-image band dimensionality for drastically simpler, more reproducible seasonal queries. This trade-off is well suited to long-term ecological and phenological analysis, but less appropriate for applications focused on high-frequency pixel-level forecasting.

Beyond temporal organization, ndvi2gif integrates complete Sentinel-1 ARD preprocessing, automated phenology metrics extraction, machine learning classification, zonal statistics, and geemap-based visualization within a unified API.

Research Impact Statement

Ndvi2Gif has demonstrated realized research impact through peer-reviewed publications, operational research infrastructure, and downstream software integration since its initial release in May 2020. The package has been applied in calibration and validation studies of Earth Observation products for phenology and surface water monitoring at the Doñana protected area (Díaz-Delgado & García-Díaz, 2024).

Ndvi2Gif serves as a foundational component of *geeltermap*, a Python mapping application providing operational environmental monitoring tools for the eLTER network (Díaz-Delgado et

al., 2024). Geeltermap integrates PhenoApp for phenology monitoring (García-Díaz & Díaz-Delgado, 2023), FloodApp for flood extent analysis, and LSTApp for land surface temperature assessment.

The package is currently used as a core analytical tool in the eLTER network infrastructure and the SUMHAL biodiversity monitoring initiative in Spanish protected areas, demonstrating readiness for reproducible multi-temporal remote sensing analysis.

AI Usage Disclosure

Ndvi2Gif was initially developed in May 2020 and underwent sustained development through 2024 before current-generation AI coding assistants became widely available. The core scientific methodology, architectural design decisions, and algorithmic implementations represent the author's original intellectual contributions. AI tools are currently used as development assistants for refactoring, documentation, debugging, and test implementation, while all strategic decisions regarding software architecture and scientific approach remain under the author's direction.

Acknowledgements

The author thanks the Google Earth Engine team and community, and especially acknowledges Qiusheng Wu for geemap, which serves as the primary foundation of Ndvi2Gif, and for his continued contributions to open-source geospatial software.

References

Díaz-Delgado, R., & García-Díaz, D. (2024). *In-situ validation of land surface phenology, land surface temperature and surface water derived from earth observation products: Doñana protected area as a potential cal/val supersite*. https://www.researchgate.net/publication/396609384

Díaz-Delgado, R., García-Díaz, D., Marangi, C., Silver, M., Oggioni, A., Tagliolato Acquaviva D'Aragona, P., Karnieli, A., Peterseil, J., Vicario, S., Anttila, S., Böttcher, K., Bolton, W., Wohner, C., & Minic, V. (2024). *eLTER PLUS deliverable D4.4 - piloting of eLTER specific downstream services contributing to eLTER standard observation variables*. eLTER PLUS Project. https://doi.org/10.5281/zenodo.14889212

García-Díaz, D., & Díaz-Delgado, R. (2023). PhenoApp: A google earth engine based tool for monitoring phenology. *Revista de Teledetección*, *61*, 73–81. https://doi.org/10.4995/raet.2023.18767

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). Google earth engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*, *202*, 18–27. https://doi.org/10.1016/j.rse.2017.06.031

Montero, D. (2021). Eemont: A python package that extends google earth engine. *Journal of Open Source Software*, *6*(62), 3168. https://doi.org/10.21105/joss.03168

Vollrath, A., Mullissa, A., & Reiche, J. (2020). Angular-based radiometric slope correction for sentinel-1 on google earth engine. *Remote Sensing*, *12*(11), 1867. https://doi.org/10.3390/rs12111867

Wohner, C., Peterseil, J., Oggioni, A., Bertrand, N., & Schentz, H. (2019). DEIMS-SDR–a web portal to document research sites and their associated data. *Ecological Informatics*, *51*, 135–140. https://doi.org/10.1016/j.ecoinf.2019.01.005

Wu, Q. (2020). Geemap: A python package for interactive mapping with google earth engine. *Journal of Open Source Software*, *5*(51), 2305. https://doi.org/10.21105/joss.02305

Zuspan, A. (2021). *Wxee: A python interface between earth engine and xarray*. https://github.com/aazuspan/wxee