

WetlandMapper: A Python package for automatic wetland mapping, dynamics classification, and cover-type characterisation from multispectral time-series data

Manudeo Singh ¹✉

¹ Department of Geography and Earth Sciences, Aberystwyth University, Aberystwyth, Wales, UK ✉
Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) ✉
- [Repository](#) ✉
- [Archive](#) ✉

Editor: [Open Journals](#) ✉

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright,
and release the work under a
Creative Commons Attribution 4.0
International License ([CC BY 4.0](#)).

Summary

WetlandMapper is an open-source Python package that operationalises two peer-reviewed remote-sensing frameworks for automated wetland analysis from multispectral satellite data, now expanded with comprehensive spectral index computation, terrain analysis, and enhanced visualisation capabilities. The package is available on PyPI (`pip install wetlandmapper`) and archived at Zenodo ([Singh, 2026](#)).

The first framework is a **wetland dynamics classification** method ([Singh & Sinha, 2022a](#)) that maps floodplain wetlands at basin scale from a multi-temporal MNDWI time series and classifies each wetland pixel into one of six temporal dynamics classes: *Persistent*, *New*, *Intensifying*, *Diminishing*, *Lost*, and *Intermittent*.

The second framework is a **Wetland Cover Type (WCT) classification** method ([Singh et al., 2022](#)) that combines multiple spectral indices to characterise the biophysical surface composition of wetland pixels into stable, ecologically interpretable cover types including open clear water, turbid water, aquatic vegetation, and moist soil.

The package now includes comprehensive **spectral index computation** supporting six indices (MNDWI, NDWI, NDVI, NDTI, AWEIsh, AWEInsh) for flexible water detection and vegetation analysis, a **terrain analysis module** for topographic corrections and artifact masking, and **visualisation utilities** for interactive plotting of results.

These modules address complementary monitoring questions: the dynamics module characterises *when and how* wetland inundation is changing over time, the WCT module characterises *what* is present at the wetland surface at any given time, the spectral indices provide flexible water and vegetation detection, and the terrain module enables topographic corrections for improved accuracy in hilly terrain. All operate on xarray DataArray objects ([Hoyer & Hamman, 2017](#)), enabling dask-backed parallel processing of large raster archives. Users may supply their own pre-processed imagery or retrieve analysis-ready surface-reflectance data directly from Google Earth Engine (GEE; Gorelick et al. ([2017](#)))) via an integrated optional acquisition submodule that supports all Landsat missions (4, 5, 7, 8, and 9), Sentinel-2, MODIS, and shapefile-path or GeoJSON area-of-interest inputs.

Statement of Need

Wetlands cover roughly 5–8% of Earth's land surface and deliver ecosystem services valued at tens of trillions of dollars annually, yet global wetland area has declined by approximately

39 35% since 1970 (Davidson, 2014; Ramsar Convention Secretariat, 2018). Operational
40 monitoring at basin to continental scales — tracking both inundation dynamics and biophysical
41 surface conditions — is essential for conservation management, restoration prioritisation, and
42 international reporting obligations under the Ramsar Convention.

43 Existing tools for remote sensing-based wetland analysis suffer from one or more of the following
44 limitations: (1) they characterise only binary inundation extent and do not resolve temporal
45 dynamics or surface cover composition (Pekel et al., 2016); (2) they are embedded in proprietary
46 platforms (ArcGIS Model Builder, GEE JavaScript API) that resist integration into scripted,
47 reproducible workflows; (3) they address either dynamics or cover characterisation but not
48 both within a single interoperable framework; (4) their methods are distributed as single-use
49 scripts rather than tested, documented software libraries with version histories; (5) they lack
50 comprehensive spectral index libraries for flexible water and vegetation detection; and (6) they
51 do not account for topographic effects that can confound wetland classification in hilly terrain.

52 WetlandMapper addresses all six gaps. It provides a fully Pythonic, open-source library that
53 unifies the dynamics-classification method of Singh & Sinha (2021) and Singh & Sinha (2022a)
54 — previously dependent on ArcGIS — and the WCT method of Singh et al. (2022) — previously
55 distributed only as GEE JavaScript code and an ArcGIS toolbox — while adding comprehensive
56 spectral index computation, terrain analysis for topographic corrections, visualisation utilities,
57 and a flexible data-ingestion pathway that supports both user-supplied imagery and automated
58 GEE retrieval. Both core methods require no labelled training data and operate on any
59 multispectral archive from which the required indices can be computed.

60 State of the Field

61 The JRC Global Surface Water dataset (Pekel et al., 2016) characterises long-term open-
62 water occurrence globally at 30 m resolution but does not distinguish wetland surface types.
63 Machine-learning approaches achieve high classification accuracy but require labelled training
64 data that are rarely available at regional scales (Mahdavi et al., 2018; Slagter et al., 2020).
65 SAR-based methods handle cloud cover but require specialist pre-processing workflows and
66 site-specific thresholding. Most tools lack comprehensive spectral index libraries or terrain
67 correction capabilities needed for robust wetland analysis across diverse landscapes.

68 The methods unified in WetlandMapper occupy a practical middle ground — no training labels,
69 applicable to any cloud-free multispectral archive, and producing ecologically interpretable
70 outputs — while providing comprehensive spectral index computation, terrain analysis for
71 topographic corrections, and visualisation utilities. WetlandMapper makes these capabilities
72 accessible in a reusable, tested Python library for the first time.

73 Software Design and Methods

74 Wetland Dynamics Classification

75 This module implements the basin-scale inventory and hydrodynamics framework of Singh &
76 Sinha (2022a). Given a multi-temporal MNDWI raster stack, each time step is first thresholded
77 to a binary water-presence layer:

$$W_t = \begin{cases} 1 & \text{if } \text{MNDWI}_t > \tau \\ 0 & \text{otherwise} \end{cases}$$

78 where $\tau = 0$ by default (positive MNDWI indicates a water-dominated pixel; Xu (2006)).
79 Three summary statistics are then derived across the full time series of length T , split by a
80 configurable temporal window n :

$$W_{\text{percent}} = \frac{\sum_{t=1}^T W_t}{T} \times 100, \quad W_{\text{historic}} = \sum_{t=1}^n W_t, \quad W_{\text{recent}} = \sum_{t=T-n+1}^T W_t$$

The temporal change signal $\Delta W = W_{\text{recent}} - W_{\text{historic}}$ is used together with W_{percent} and two user-adjustable wet-frequency thresholds (θ_{wet} , θ_{persis}) to assign each pixel to one of six dynamics classes:

Class	Primary condition
Persistent	$W_{\text{percent}} \geq \theta_{\text{persis}}$
New	$\Delta W = +n$
Intensifying	$W_{\text{percent}} \geq \theta_{\text{wet}}; 0 < \Delta W < n$
Diminishing	$W_{\text{percent}} \geq \theta_{\text{wet}}; -n < \Delta W < 0$
Lost	$\Delta W = -n$
Intermittent	$W_{\text{percent}} \geq \theta_{\text{wet}}; \text{no directional trend}$
Non-wetland	$W_{\text{percent}} < \theta_{\text{wet}}$

The entire classification is implemented using vectorised `xr.where` operations, enabling chunked parallel execution via Dask with no Python-level pixel iteration.

Wetland Cover Type Classification

This module implements the multi-index WCT framework of Singh et al. (2022). Three spectral indices are computed from the same multispectral image:

$$\text{MNDWI} = \frac{G - \text{SWIR}}{G + \text{SWIR}}, \quad \text{NDVI} = \frac{\text{NIR} - R}{\text{NIR} + R}, \quad \text{NDTI} = \frac{R - G}{R + G}$$

MNDWI delineates the extent of surface water; NDVI quantifies vegetation presence and density; NDTI quantifies water turbidity. Their combined spectral signatures partition wetland pixels into five biophysically distinct cover types. Two classification implementations are provided: the original quartile-based combination code method (`classify_wct_ema`), and an improved continuous-threshold variant (`classify_wct`) that allows sub-quartile calibration for different sensors or seasons.

Spectral Index Computation

The package provides comprehensive spectral index computation supporting six indices for flexible water detection and vegetation analysis:

- **MNDWI**: Modified Normalised Difference Water Index (Xu, 2006)
- **NDWI**: Normalised Difference Water Index (McFeeters, 1996)
- **NDVI**: Normalised Difference Vegetation Index
- **NDTI**: Normalised Difference Turbidity Index
- **AWEIsh**: Automated Water Extraction Index with shadow suppression (Feyisa et al., 2014)
- **AWEInsh**: Automated Water Extraction Index without shadow suppression (Feyisa et al., 2014)

Users can compute individual indices or use `compute_indices()` for the core WCT indices (MNDWI, NDVI, NDTI) or `compute_water_indices()` for comprehensive water detection across all available water indices.

109 Terrain Analysis

110 The terrain analysis module provides topographic corrections essential for accurate wetland
111 classification in hilly or mountainous terrain where slope and topographic position can confound
112 spectral signatures. Functions include:

- 113 ■ **Slope computation:** Local slope calculation from digital elevation models
- 114 ■ **Topographic Position Index (TPI):** Relative topographic position within a local
115 neighborhood
- 116 ■ **Local range:** Local elevation variability for terrain complexity assessment
- 117 ■ **DEM depression mapping:** Identification of closed depressions from raw versus pit-filled
118 DEMs using integer division and binary reclassification, following the riverine wetland
119 mapping protocol of Sinha et al. (2017)
- 120 ■ **Terrain artifact masking:** Automated masking of steep slopes that may cause classification
121 errors in wetland detection

122 This terrain functionality therefore supports both topographic false-positive reduction in rugged
123 settings and depression-focused wetland candidate mapping in low-relief floodplains.

124 Data Acquisition and Temporal Aggregation

125 The optional `wetlandmapper.gee` submodule supports all five Landsat missions (4, 5, 7, 8,
126 9), Sentinel-2, and MODIS. A "LandsatAll" option automatically merges available missions
127 for any requested date range with harmonised band names, enabling long-record analyses
128 from 1982 to the present day. "MODISAll" provides similar functionality for MODIS Terra
129 and Aqua missions. An optional `use_slc_off` parameter controls whether Landsat 7 images
130 acquired after the 2003 Scan Line Corrector failure (which cause ~22% data gaps per scene)
131 are included. Areas of interest may be provided as a GeoJSON dict, a shapefile path, or a
132 GeoJSON file path; multi-feature shapefiles are dissolved to a single boundary automatically.

133 Server-side temporal compositing reduces data transfer volume for long time series: users may
134 request one median composite per year, month, or meteorological season (DJF / MAM / JJA
135 / SON) rather than every individual scene. The same temporal aggregation functionality is
136 also available client-side via `aggregate_time()`, which operates on any xarray DataArray or
137 Dataset regardless of data source.

138 Dependencies and Installation

139 WetlandMapper requires Python 3.9. Core dependencies are numpy (Harris et al., 2020), xarray
140 (Hoyer & Hamman, 2017), and rioxarray (Snow & others, 2022). The GEE submodule
141 additionally requires earthengine-api, rasterio, xee, dask, and geopandas. The plotting
142 submodule requires matplotlib (Hunter, 2007).

143 Installation via **conda** (recommended for ease of dependency management):

```
# Core functionality
```

```
conda install -c conda-forge wetlandmapper
```

```
# With Google Earth Engine support
```

```
conda install -c conda-forge wetlandmapper geopandas earthengine-api xee dask rasterio
```

```
# Complete installation with all extras
```

```
conda install -c conda-forge wetlandmapper geopandas earthengine-api xee dask rasterio m
```

144 Alternatively, installation via **pip**:

```
pip install wetlandmapper
```

```
# core
```

```
pip install "wetlandmapper[gee]"
```

```
# with GEE + shapefile support
```

```
pip install "wetlandmapper[plot]"          # with visualisation utilities
pip install "wetlandmapper[all]"          # complete installation
```

145 Detailed platform-specific installation instructions, including GEE authentication setup, are
146 provided in `INSTALL.md` in the repository.

147 Usage

148 A minimal end-to-end example for each workflow:

```
import xarray as xr
from wetlandmapper import compute_mndwi, classify_dynamics
from wetlandmapper import compute_indices, classify_wct_ema
from wetlandmapper import compute_water_indices, compute_slope
from wetlandmapper.plotting import plot_dynamics, plot_wct
from wetlandmapper.gee import fetch

# --- Dynamics: fetch annual composites from all Landsat missions ---
mndwi = fetch("study_area.shp", "1984-01-01", "2023-12-31",
              sensor="LandsatAll", temporal_aggregation="annual")
dynamics = classify_dynamics(mndwi, nYear=3,
                             thresholdWet=25, thresholdPersis=75)
dynamics.rio.to_raster("wetland_dynamics.tif")

# --- WCT: single composite → 5 biophysical cover types
# --- ds_composite is an xarray multispectral dataset (a Landsat .tiff)
ds_composite = xr.load_dataset('Landsat.tiff')
indices = compute_indices(ds_composite, green_band="B3", red_band="B4",
                          nir_band="B5", swir_band="B6")
wct = classify_wct_ema(indices)
wct.rio.to_raster("wetland_cover_types.tif")

# --- Comprehensive water detection with all available indices ---
water_indices = compute_water_indices(ds_composite,
                                       blue_band="B2", green_band="B3",
                                       red_band="B4", nir_band="B5",
                                       swir_band="B6", swir2_band="B7")
water_indices # Contains MNDWI, NDWI, AWEIsh, AWEInsh

# --- Terrain analysis for topographic corrections ---
dem = xr.open_dataset("elevation.nc")["elevation"]
slope = compute_slope(dem)
# Apply terrain masking to reduce false positives in steep areas

# --- Visualisation ---
plot_dynamics(dynamics, title="Wetland Dynamics Classification")
plot_wct(wct, title="Wetland Cover Types")
```

149 A Jupyter notebook demonstrating both workflows on synthetic data, with full GEE acquisition,
150 terrain analysis, and interactive visualisation sections, is included in the repository.

151 Validation and Results

152 The dynamics module is derived from Singh & Sinha (2022a), where the frequency-based
153 temporal aggregation method can effectively distinguish permanently inundated, seasonally

active, and recently changed wetlands at basin scale, providing a validated inventory framework applicable to wetland restoration prioritisation (Singh & Sinha, 2022b).

The WCT module is derived from Singh et al. (2022), where the MNDWI–NDVI–NDTI combination was validated across three Ramsar-listed wetlands in contrasting geomorphic and climatic settings — Kaabar Tal (Ganga floodplain), Chilika Lagoon (coastal), and Nal Sarovar (semi-arid) — demonstrating that the WCTs are stable in space and time, meaning comparable biophysical conditions correspond to the same WCT irrespective of geographic location.

The depression-mapping terrain workflow follows the riverine wetland protocol of Sinha et al. (2017), in which raw and pit-filled DEMs are compared to delineate closed depressions that can act as wetland candidates in flat floodplain environments.

By unifying both workflows in a single, tested Python package with flexible data ingestion, WetlandMapper enables application of these frameworks globally in fully scripted, reproducible workflows, without dependence on proprietary GEE or ArcGIS environments.

Acknowledgements

The author acknowledges the ISRO–IIT Kanpur Space Technology Cell and WWF-India for funding the original research underlying both methods, and the Google Earth Engine team for providing cloud-computing access. The author is a Newton International Fellow funded by The Royal Society, London, and gratefully acknowledges this support. The facilities and support provided by Aberystwyth University are also duly acknowledged.

AI Assistance

Development of this software package was assisted by Claude (Anthropic), an AI language model, for tasks including code scaffolding, packaging configuration, continuous integration setup, docstring writing, and debugging. The scientific methodology, classification algorithms, spectral index thresholds, and validation are based entirely on the author's prior peer-reviewed work (Singh et al., 2022; Singh & Sinha, 2021, 2022a; Sinha et al., 2017), and all scientific content, design decisions, and results presented here are the author's own.

References

- Davidson, N. C. (2014). How much wetland has the world lost? Long-term and recent trends in global wetland area. *Marine and Freshwater Research*, 65(10), 934–941. <https://doi.org/10.1071/MF14173>
- Feyisa, G. L., Meilby, H., Fensholt, R., & Proud, S. R. (2014). Automated water extraction index: AWE for surface water mapping AW3D. *Remote Sensing of Environment*, 140, 23–35. <https://doi.org/10.1016/j.rse.2013.08.029>
- 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>
- Harris, C. R., Millman, K. J., Walt, S. J. van der, & others. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362. <https://doi.org/10.1038/s41586-020-2649-2>
- Hoyer, S., & Hamman, J. J. (2017). Xarray: N-D labeled arrays and datasets in Python. *Journal of Open Research Software*, 5(1), 10. <https://doi.org/10.5334/jors.148>
- Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95. <https://doi.org/10.1109/MCSE.2007.55>

- 196 Mahdavi, S., Salehi, B., Granger, J., Amani, M., Brisco, B., & Huang, W. (2018). Remote
197 sensing for wetland classification: A comprehensive review. *GIScience & Remote Sensing*,
198 55(5), 623–658. <https://doi.org/10.1080/15481603.2017.1419602>
- 199 McFeeters, S. K. (1996). The use of the Normalized Difference Water Index (NDWI) in
200 the delineation of open water features. *International Journal of Remote Sensing*, 17(7),
201 1425–1432. <https://doi.org/10.1080/01431169608948714>
- 202 Pekel, J.-F., Cottam, A., Gorelick, N., & Belward, A. S. (2016). High-resolution mapping
203 of global surface water and its long-term changes. *Nature*, 540(7633), 418–422. <https://doi.org/10.1038/nature20584>
- 204 Ramsar Convention Secretariat. (2018). *Global wetland outlook: State of the world's wetlands
205 and their services to people*. Ramsar Convention Secretariat. [https://www.global-wetland-
206 outlook.ramsar.org](https://www.global-wetland-outlook.ramsar.org)
- 207 Singh, M. (2026). *WetlandMapper: Automatic wetland mapping, dynamics classification, and
208 cover-type characterisation from multispectral time-series data* (Version v1.1.3). Zenodo.
209 <https://doi.org/10.5281/zenodo.18967177>
- 210 Singh, M., Allaka, S., Gupta, P. K., Patel, J. G., & Sinha, R. (2022). Deriving wetland-cover
211 types (WCTs) from integration of multispectral indices based on Earth observation data.
212 *Environmental Monitoring and Assessment*, 194(12), 878. [https://doi.org/10.1007/s10661-
213 022-10541-7](https://doi.org/10.1007/s10661-022-10541-7)
- 214 Singh, M., & Sinha, R. (2021). Hydrogeomorphic indicators of wetland health inferred from
215 multi-temporal remote sensing data for a new Ramsar site (kaabar tal), India. *Ecological
216 Indicators*, 127(1), 107739. <https://doi.org/10.1016/j.ecolind.2021.107739>
- 217 Singh, M., & Sinha, R. (2022a). A basin-scale inventory and hydrodynamics of floodplain
218 wetlands based on time-series of remote sensing data. *Remote Sensing Letters*, 13(1),
219 1–13. <https://doi.org/10.1080/2150704X.2021.1980919>
- 220 Singh, M., & Sinha, R. (2022b). Integrating hydrological connectivity in a process–response
221 framework for restoration and monitoring prioritisation of floodplain wetlands in the
222 Ramganga basin, India. *Water*, 14(21), 3520. <https://doi.org/10.3390/w14213520>
- 223 Sinha, R., Saxena, S., & Singh, M. (2017). Protocols for riverine wetland mapping and
224 classification using remote sensing and GIS. *Current Science*, 112(7), 1544–1552. [http://
225 www.jstor.org/stable/24912702](http://www.jstor.org/stable/24912702)
- 226 Slagter, B., Tsendbazar, N.-E., Vollrath, A., & Reiche, J. (2020). Mapping wetland
227 characteristics using temporally dense Sentinel-1 and Sentinel-2 data: A case study in the
228 St. Lucia wetlands, South Africa. *International Journal of Applied Earth Observation and
229 Geoinformation*, 86, 102009. <https://doi.org/10.1016/j.jag.2019.102009>
- 230 Snow, A. D., & others. (2022). *Rioxarray: Geospatial xarray extension powered by rasterio*.
231 Zenodo. <https://doi.org/10.5281/zenodo.4570456>
- 232 Xu, H. (2006). Modification of normalised difference water index (NDWI) to enhance open
233 water features in remotely sensed imagery. *International Journal of Remote Sensing*, 27(14),
234 3025–3033. <https://doi.org/10.1080/01431160600589179>
- 235