




# Leafmap: A Python package for interactive mapping and geospatial analysis with minimal coding in a Jupyter environment

Qiusheng Wu<sup>1</sup>

<sup>1</sup> Department of Geography, University of Tennessee, Knoxville, TN 37996, United States

DOI: [10.21105/joss.03414](https://doi.org/10.21105/joss.03414)

## Software

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

Editor: [Andrew Stewart](#) 

## Reviewers:

- [@martinfleis](#)
- [@TomasBeuzen](#)

Submitted: 23 June 2021

Published: 26 July 2021

## License

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

## Summary

**Leafmap** is a Python package for interactive mapping and geospatial analysis with minimal coding in a Jupyter environment. It is a spin-off project of the [geemap](#) Python package ([Wu, 2020](#)), which was designed specifically to work with [Google Earth Engine](#) (GEE) ([Gorelick et al., 2017](#)). However, not everyone in the geospatial community has access to the GEE cloud computing platform. Leafmap is designed to fill this gap for non-GEE users. It is a free and open-source Python package that enables users to analyze and visualize geospatial data with minimal coding in a Jupyter environment, such as Google Colab, Jupyter Notebook, and JupyterLab. Leafmap is built upon several open-source packages, such as [folium](#) ([Filipe et al., 2021](#)) and [ipyleaflet](#) ([Renou et al., 2021](#)) (for creating interactive maps), [WhiteboxTools](#) ([Lindsay, 2018](#)) and [whiteboxgui](#) (for analyzing geospatial data), and [ipywidgets](#) ([Grout & et al., 2021](#)) (for designing interactive graphical user interface [GUI]). Leafmap has a toolset with various interactive tools that allow users to load vector and raster data onto the map without coding. In addition, users can use the powerful analytical backend (i.e., WhiteboxTools) to perform geospatial analysis directly within the leafmap user interface without writing a single line of code. The WhiteboxTools library currently contains **468** tools for advanced geospatial analysis, such as [GIS Analysis](#), [Geomorphometric Analysis](#), [Hydrological Analysis](#), [LiDAR Data Analysis](#), [Mathematical and Statistical Analysis](#), and [Stream Network Analysis](#).

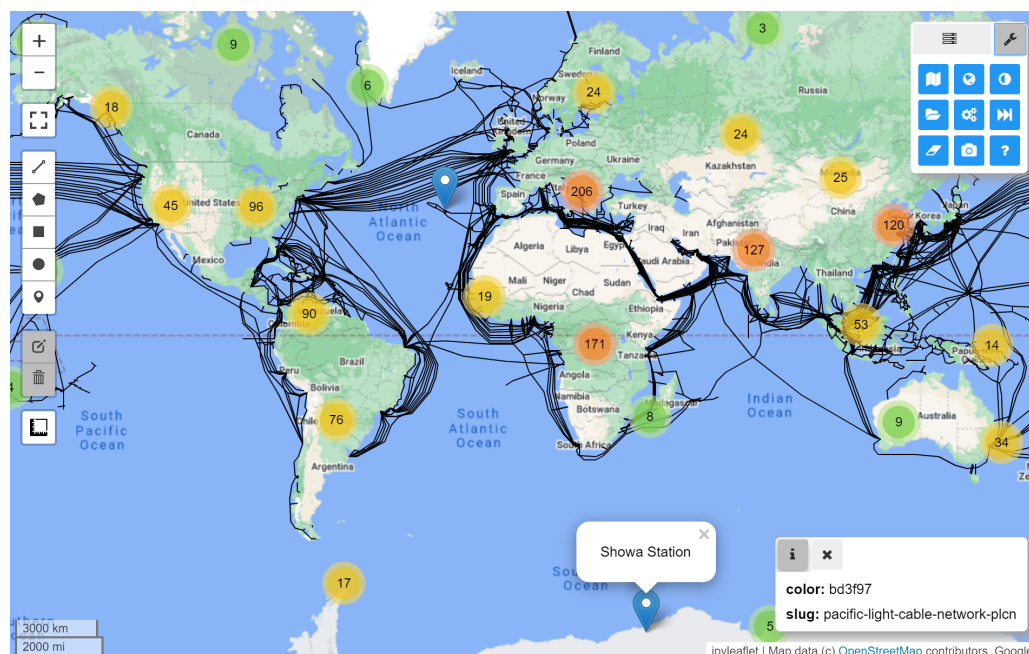
## Statement of Need

There is a plethora of Python packages for geospatial analysis, such as [geopandas](#) for vector data analysis ([Jordahl et al., 2021](#)) and [xarray](#) for raster data analysis ([Hoyer & Hamman, 2017](#)). However, few Python packages provide interactive GUIs for loading geospatial data in a Jupyter environment. It might take many lines of code to load and display geospatial data with various file formats on an interactive map, which can be a challenging task for novice users with limited coding skills. There are also some notable Python packages for visualizing geospatial data in a Jupyter environment, such as [plotly](#) ([Mease & et al., 2021](#)) and [kepler.gl](#) ([He & et al., 2021](#)). However, plotly is designed for displaying static data, which lacks bidirectional communication between the front-end and the backend. Kepler.gl provides unique 3D functionality for visualizing large-scale geospatial datasets, but it lacks tools for performing geospatial analysis, such as hydrological analysis and LiDAR data analysis. In contrast, leafmap provides many convenient functions for loading and visualizing geospatial datasets with only one line of code. Users can also use the interactive GUI to load geospatial datasets without coding. Leafmap is intended for anyone who would like to analyze and visualize geospatial data interactively in a Jupyter environment. It is particularly suited for novice users with limited programming skills. Advanced programmers can also find leafmap a useful tool for analyzing geospatial data and building interactive web apps.

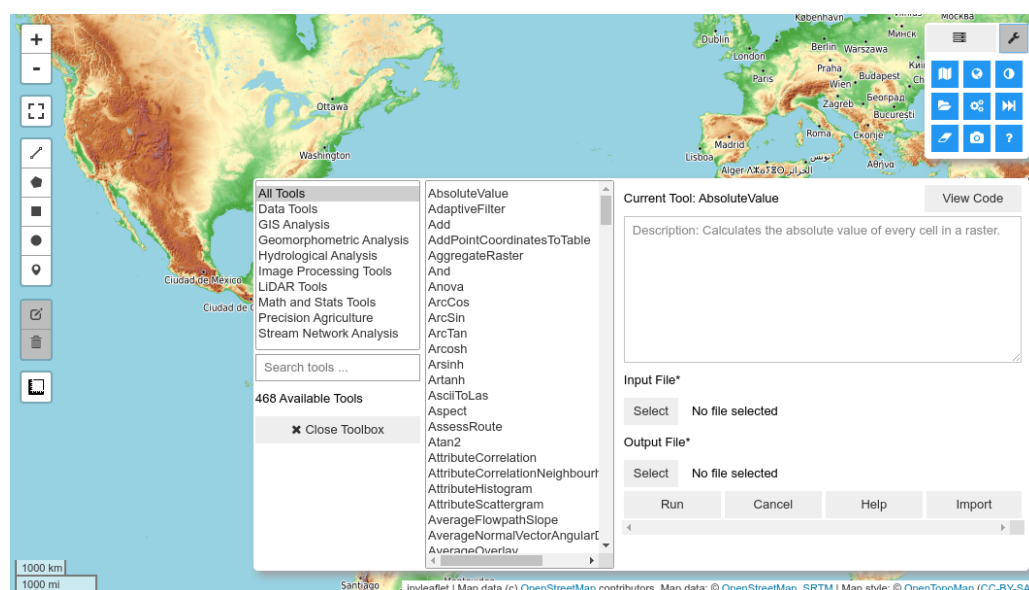
## Leafmap Plotting Backends

**Leafmap** has three plotting backends, including [folium](#), [ipyleaflet](#), and [here-map-widget-for-jupyter](#) (Kharude & Steenbergen, 2021). An interactive map created using one of the plotting backends can be displayed in a Jupyter environment, such as Google Colab, Jupyter Notebook, and JupyterLab. By default, `import leafmap` in [Jupyter Notebook](#) and [JupyterLab](#) will use the `ipyleaflet` plotting backend, whereas `import leafmap` in [Google Colab](#) will use the `folium` plotting backend. Note that Google Colab does not yet support custom widgets, such as `ipyleaflet` and `heremap` widget ([source](#)). Therefore, interactive maps created using the `ipyleaflet` and `heremap` widget backends won't show up in Google Colab, even though the code might run successfully without any errors.

The three plotting backends do not offer equal functionality. The `ipyleaflet` plotting backend provides the richest interactive functionality, including the custom toolset for loading, analyzing, and visualizing geospatial data interactively without coding. For example, users can add vector data (e.g., GeoJSON, Shapefile, KML, GeoDataFrame) and raster data (e.g., GeoTIFF, Cloud Optimized GeoTIFF [COG]) to the map with a few clicks (see Figure 1). Users can also perform geospatial analysis using the WhiteboxTools GUI with 468 geoprocessing tools directly within the map interface (see Figure 2). Other interactive functionality (e.g., split-panel map, linked map, time slider, time-series inspector) can also be useful for visualizing geospatial data. The `ipyleaflet` package is built upon `ipywidgets` and allows bidirectional communication between the front-end and the backend enabling the use of the map to capture user input ([source](#)). In contrast, `folium` has relatively limited interactive functionality. It is meant for displaying static data only. The `folium` plotting backend is included in this package to support using `leafmap` in Google Colab. Note that the aforementioned custom toolset and interactive functionality are not available for the `folium` plotting backend. Compared with `ipyleaflet` and `folium`, the `heremap` widget plotting backend provides some unique [3D functionality](#) for visualizing geospatial data. An [API key](#) from the [Here Developer Portal](#) is required to use `heremap`.



**Figure 1.** The leafmap user interface built upon `ipyleaflet` and `ipywidgets`.



**Figure 2.** The WhiteboxTools graphical user interface integrated into leafmap.

## Leafmap Modules

The key functionality of the leafmap Python package is organized into nine modules as shown in the table below.

Module	Description
basemaps	A collection of XYZ and WMS tile layers to be used as basemaps
colormaps	Commonly used colormaps and palettes for visualizing geospatial data
common	Functions being used by multiple plotting backends to process geospatial data
foliummap	A plotting backend built upon the folium Python package
heremap	A plotting backend built upon the here-map-widget-for-jupyter
leafmap	The default plotting backend built upon the ipyleaflet Python package
legends	Built-in legends for commonly used geospatial datasets
osm	Functions for extracting and downloading OpenStreetMap data
toolbar	A custom toolset with interactive tools built upon ipywidgets and ipyleaflet

## Leafmap Tutorials

Comprehensive documentation and API reference of the leafmap package is available at <https://geemap.org>. A list of notebook examples and video tutorials for using leafmap can be found at <https://leafmap.org/tutorials>. Users can also try out leafmap using Google Colab (<https://github.com/leafmap-colab>) and Binder (<https://github.com/leafmap-pangeo>) using an Internet browser without having to set up the Python environment and install leafmap on their computer.

## Acknowledgements

The author would like to thank the developers of ipyleaflet and ipywidgets, which empower the interactive mapping functionality of leafmap, including [Martin Renou](#), [David Brochart](#), and

Sylvain Corlay. The authors would also like to express thanks to [John Lindsay](#) for developing the WhiteboxTools library, which serves as the geospatial analysis backend of leafmap. Special thanks go to all leafmap contributors, especially [Sachin Kharude](#) for contributing the heremap plotting backend to leafmap. Last but not least, the author would like to thank [Tomas Beuzen](#) and [Martin Fleischmann](#) for reviewing this paper and the leafmap package. Their constructive comments greatly improved the quality of the source code and documentation of the leafmap package as well as this paper.

## References

- Filipe, Journois, M., Frank, Story, R., Gardiner, J., Rump, H., Bird, A., Lima, A., Cano, J., dbf, Leonel, J., Baker, J., Sampson, T., Reades, J., Welsh, B., Kong, Q., Komarov, O., Crosby, A., Harris, G., ... Signell, R. (2021). *python-visualization/folium v0.12.1*. <https://doi.org/10.5281/zenodo.4447642>
- 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>
- Grout, J., & et al. (2021). *ipywidgets: Interactive HTML Widgets*. Github. <https://github.com/jupyter-widgets/ipywidgets>
- He, S., & et al. (2021). *kepler.gl: A powerful open source geospatial analysis tool for large-scale data sets*. Github. <https://github.com/kepler.gl/kepler.gl>
- Hoyer, S., & Hamman, J. (2017). xarray: ND labeled arrays and datasets in Python. *Journal of Open Research Software*, 5(1). <https://doi.org/10.5334/jors.148>
- Jordahl, K., Van den Bossche, J., Fleischmann, M., McBride, J., Wasserman, J., & Gerard, J. (2021). *geopandas/geopandas: v0.9.0*. <https://doi.org/10.5281/zenodo.4569086>
- Kharude, S., & Steenbergen, T. (2021). *here-map-widget-for-jupyter*. Github. <https://github.com/heremaps/here-map-widget-for-jupyter>
- Lindsay, J. B. (2018). *WhiteboxTools User Manual*. [https://jblindsay.github.io/wbt\\_book](https://jblindsay.github.io/wbt_book)
- Mease, J., & et al. (2021). *plotly.py: The interactive graphing library for Python*. Github. <https://github.com/plotly/plotly.py>
- Renou, M., Corlay, S., Brochart, D., & et al. (2021). *ipyleaflet: A Jupyter / Leaflet bridge enabling interactive maps in the Jupyter notebook*. Github. <https://github.com/jupyter-widgets/ipyleaflet>
- Wu, Q. (2020). geemap: A Python package for interactive mapping with Google Earth Engine. *The Journal of Open Source Software*, 5(51), 2272. <https://doi.org/10.21105/joss.02272>