The Journal of Open Source Software

BatchPlanet: Batch access and processing of PlanetScope imagery for spatiotemporal analysis in R

Yiluan Song [1,2], **Alexandre R. Bevington** [3,4], **Ken Reid** [1], **Yi Liu** [2], **Jiali Zhu** [2], **and Kai Zhu** [2]

1 Michigan Institute for Data and AI in Society, University of Michigan, Ann Arbor, MI, USA **2** School for Environment and Sustainability and Institute for Global Change Biology, University of Michigan, Ann Arbor, MI, USA **3** Department of Geography, Earth and Environmental Sciences, University of Northern British Columbia, Prince George, BC, Canada **4** Ministry of Forests, Province of British Columbia, Prince George, BC, Canada

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: William Gearty

Reviewers:

- @ajpelu
- @QuinnAsena

Submitted: 25 July 2025
Published: unpublished

Summary

The `BatchPlanet` R package offers a reproducible and scalable workflow for accessing and processing PlanetScope satellite imagery. It is designed for environmental researchers to efficiently perform spatiotemporal analysis on high-resolution remote sensing data. The package streamlines imagery ordering and downloading with the PlanetScope API, retrieval and cleaning of pixel-level time series, and computation of derived metrics like the Enhanced Vegetation Index (EVI) and phenological metrics (e.g., green-up/down time). This tool is particularly valuable for research involving large volumes of imagery across multiple sites and extended time periods.

Statement of Need

PlanetScope imagery provides global, high-resolution (~3-meter), near-daily data, making it valuable for scientific research and monitoring of phenology (Moon et al., 2021), land use change, disaster impacts, and more. While the Planet API facilitates access (Team, 2017), using this data remains challenging due to complex API interactions, limits on large-volume data downloads, and non-trivial processing workflows, which can hinder reproducibility.

Existing options include the official Planet Python SDK, cloud-based platforms like Sentinel Hub and Google Earth Engine (GEE), and the R package `planetR`. However, the Python and JavaScript used by the first three platforms may be less familiar to R users. Cloud platforms also restrict user control over processing environments and local data storage. Furthermore, existing tools typically require users to write extensive custom scripts for batch downloading and processing across multiple sites, a process often limited by computing and storage resources. `BatchPlanet` addresses these gaps by providing an R-native tool for batch access and processing of PlanetScope imagery (Table 1). Its streamlined, parallelized functions are designed for scalability, transparency, and reproducibility in scientific data pipelines. This package has supported peer-reviewed research in predicting reproductive phenology in wind-pollinated trees (Song et al., 2025).

Table 1: Table 1. Comparison of `BatchPlanet` with existing tools for PlanetScope data access and processing.

Feature / Tool	**Planet Python SDK**	**Sentinel Hub**	**Google Earth Engine (GEE)**	**planetR (Bevington)**	**BatchPlanet**
Primary Language	Python	Python	JavaScript / Python	R	R
Processing Environment	Local/Cloud	Cloud	Cloud	Local	Local
Data Control & Reproducibility	High	Moderate	Low	High	High
Batch Processing	Via scripting/CLI	Supported for enterprise users	Via scripting	Via scripting	Streamlined
Time Series Analysis Tools	Not supported	Limited	Supported	Not supported	Supported
Interactive Visualization	Not supported	Limited	Supported	Not supported	Supported

Key Features

The package is tailored for researchers and practitioners who: - Conduct time series analyses across spatially dispersed monitoring sites. - Work primarily in R and seek alternatives to Python-based tools. - Prioritize reproducibility in remote sensing workflows. - Use high-performance computing (HPC) infrastructure. - Need interactive visualization of PlanetScope imagery and processed data products.

A major hurdle in using PlanetScope data is the volume of data and the risk of hitting Planet API rate limits during mass downloads. This could happen, for example, when a phenological study requires years of time series over multiple locations. `BatchPlanet` provides solutions for streamlined batch ordering and downloading. It searches for images by month and automatically splits large requests into smaller orders, ensuring complete data retrieval without hitting API rate limits. It enables users to specify multiple, spatially distant sites, allowing for efficient, parallelized downloading of only the relevant imagery, minimizing unnecessary data volume.

In addition to ordering and downloading, `BatchPlanet` facilitates the entire R-native workflow for PlanetScope imagery processing, with a focus on temporal analysis. These include functions to retrieve pixel-level time series data, clean reflectance time series, calculate the Enhanced Vegetation Index (EVI), and compute phenological metrics such as green-up and green-down dates. Apart from the streamlined batch processing functions, `BatchPlanet` provides individual functions for key steps of the workflow, allowing users to customize their data processing pipelines. `BatchPlanet` also enables interactive visualization of true color images and EVI time series (Fig. 1, 2).

These batch functions significantly accelerate the process. For example, images for an approximately 9 km^2 area over one month were downloaded in 6.7 seconds. The total downloading time for a year's worth of images is comparable due to parallelization across months. Time series retrieval (reflectances, quality masks, and metadata) for 100 coordinates from one month of downloaded images took only 20.9 seconds, which can be parallelized across sites and coordinate groups.

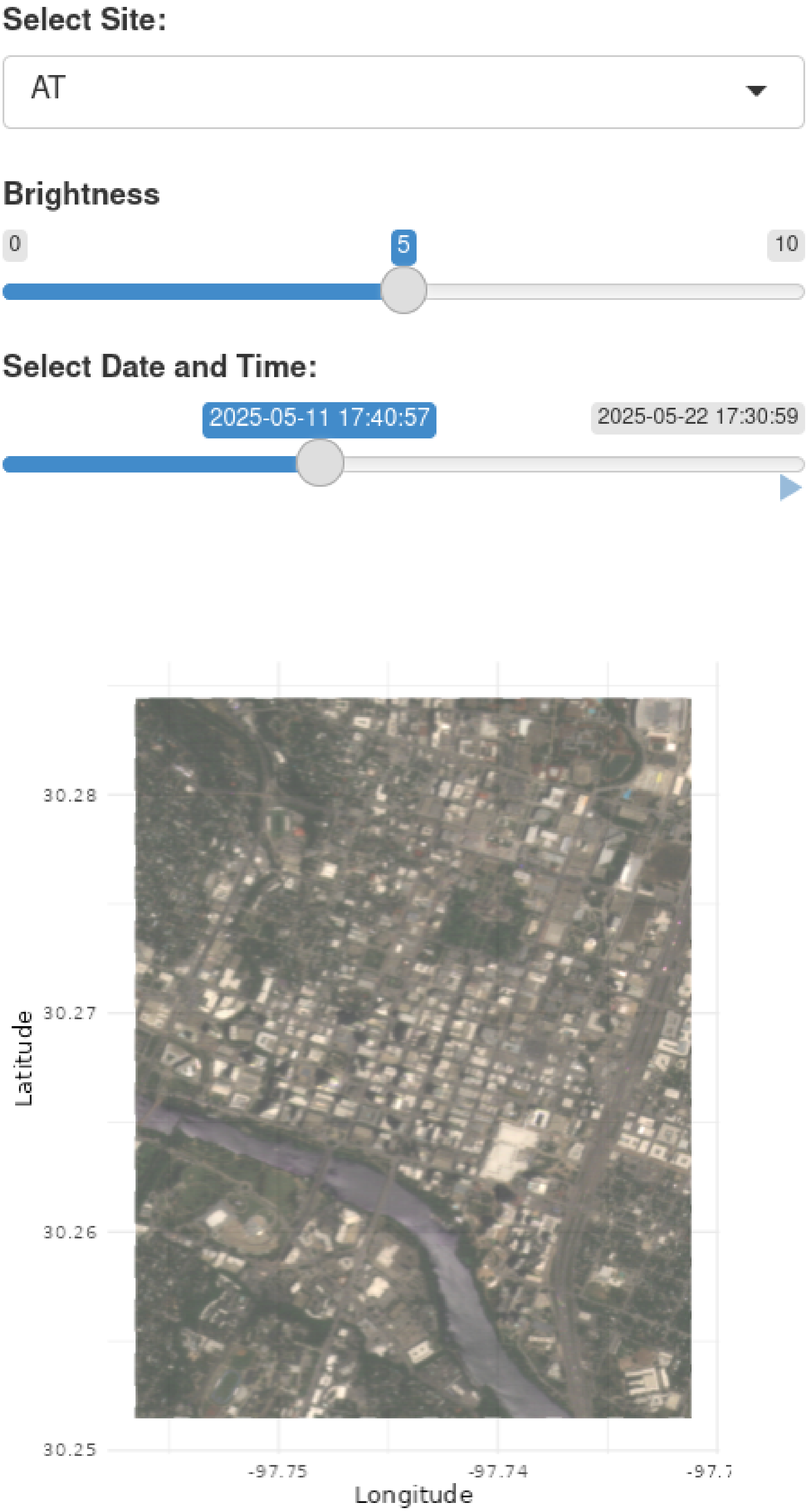

Figure 1: **Figure 1.** A screenshot of the interactive PlanetScope imagery viewer in the `BatchPlanet` package, showing a true color image in part of Austin, USA, captured on May 21, 2025.

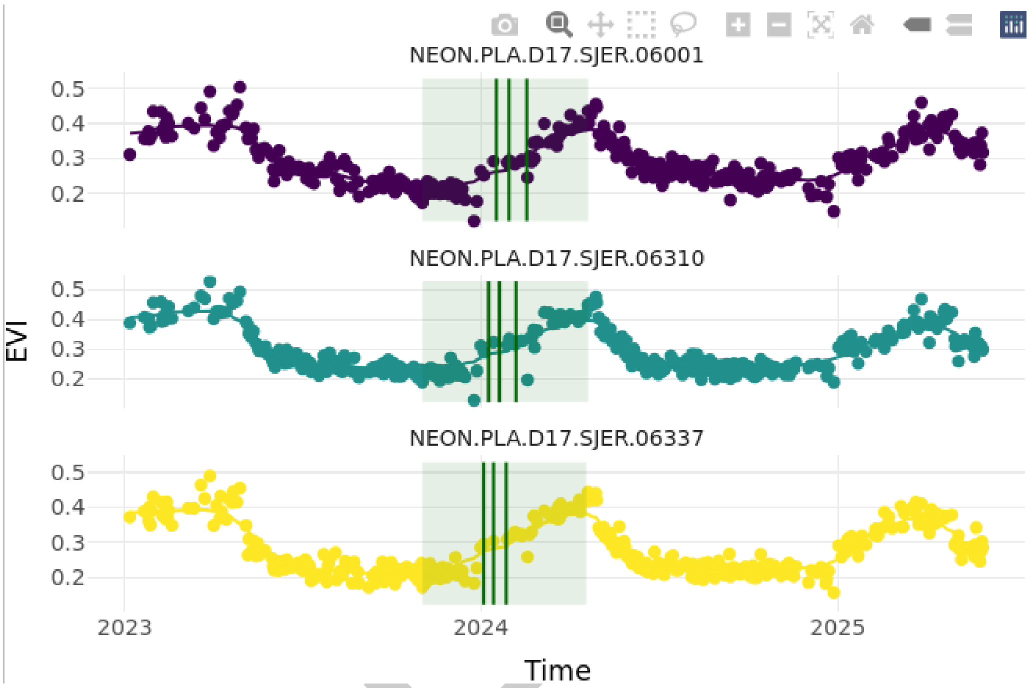

Figure 2: Figure 2. Enhanced Vegetation Index (EVI) for three trees in San Joaquin Experimental range (SJER) NEON site with phenological metrics annotated, with indices and metrics calculated and visualized using the `BatchPlanet` package.

Example Usage

Install package in R and load package.

```
install.packages('BatchPlanet',
  repos = c('https://yiluansong.r-universe.dev',
            'https://cloud.r-project.org'))
library(BatchPlanet)
```

Read example coordinates.

```
df_coordinates <- read.csv(
  system.file("extdata/NEON/example_neon_coordinates.csv",
              package = "BatchPlanet")
)
visualize_coordinates(df_coordinates)
```

Set download parameters and data directory.

```
setting <- set_planetscope_parameters(
  api_key = set_api_key(),
  item_name = "PSScene",
  asset = "ortho_analytic_4b_sr",
  product_bundle = "analytic_sr_udm2",
  cloud_lim = 0.3,
  harmonized = TRUE
)

download_sample_data()
dir_data <- "sample-data"
```

The Journal of Open Source Software

```r
dir_data_NEON <- file.path(dir_data, "NEON")
```

Order and download imagery.

```r
order_planetscope_imagery_batch(
  dir = dir_data_NEON,
  df_coordinates = df_coordinates,
  v_site = c("HARV", "SJER"),
  v_year = 2024,
  setting = setting
)

download_planetscope_imagery_batch(
  dir = dir_data_NEON,
  setting = setting,
  num_cores = 12
)

visualize_true_color_imagery_batch(
  dir = dir_data_NEON,
  df_coordinates = df_coordinates
)
```

Retrieve time series at coordinates of interest.

```r
retrieve_planetscope_time_series_batch(
  dir = dir_data_NEON,
  df_coordinates = df_coordinates,
  num_cores = 12
)

df_ts <- read_data_product(
  dir = dir_data_NEON,
  product_type = "ts"
)

visualize_time_series(
  df_ts = df_ts,
  var = "green",
  ylab = "Green reflectance",
  facet_var = "site",
  smooth = F)
```

Clean time series and calculate EVI.

```r
clean_planetscope_time_series_batch(
  dir = dir_data_NEON,
  num_cores = 3,
  calculate_evi = T
)

df_clean <- read_data_product(
  dir = dir_data_NEON,
  product_type = "clean"
)

visualize_time_series(
  df_ts = df_clean,
```

The Journal of Open Source Software

```r
  var = "evi",
  ylab = "EVI",
  facet_var = "site",
  smooth = T
)
```

Calculate phenological metrics.

```r
df_thres <- set_thresholds(
  thres_up = c(0.3, 0.4, 0.5),
  thres_down = NULL
)

calculate_phenological_metrics_batch(
  dir = dir_data_NEON,
  v_site = "SJER",
  v_group = "Quercus",
  df_thres = df_thres,
  var_index = "evi",
  num_cores = 12
)

v_id <- c("NEON.PLA.D17.SJER.06001",
          "NEON.PLA.D17.SJER.06337",
          "NEON.PLA.D17.SJER.06310")
df_doy <- read_data_product(
  dir = dir_data_NEON,
  product_type = "doy"
)
df_doy_sample <- df_doy[df_doy$id %in% v_id, ]
df_evi <- read_data_product(
  dir = dir_data_NEON,
  product_type = "clean"
)
df_evi_sample <- df_evi[df_evi$id %in% v_id, ]
visualize_time_series(
  df_ts = df_evi_sample,
  df_doy = df_doy_sample,
  var = "evi",
  ylab = "EVI",
  facet_var = "id",
  smooth = T
)
```

Acknowledgements

Yiluan Song was supported by the Eric and Wendy Schmidt AI in Science Postdoctoral Fellowship, a Schmidt Sciences program. Kai Zhu and Yiluan Song were supported by the National Science Foundation [grant numbers 2306198 (CAREER)]. We thank the Planet team for providing access to their API.

References

Moon, M., Richardson, A. D., & Friedl, M. A. (2021). Multiscale assessment of land surface phenology from harmonized landsat 8 and sentinel-2, PlanetScope, and PhenoCam imagery. *Remote Sensing of Environment*, *266*, 112716. https://doi.org/10.1016/j.rse.2021.112716

Song, Y., Katz, D. S., Zhu, Z., Beaulieu, C., & Zhu, K. (2025). Predicting reproductive phenology of wind-pollinated trees via PlanetScope time series. *Science of Remote Sensing*, 100205. https://doi.org/10.1016/j.srs.2025.100205

Team, P. (2017). *Planet application program interface: In space for life on earth*. https://api.planet.com.