Package 'land4health'

November 10, 2026

Title Remote Sensing Metrics for Spatial Health Analysis

Version 0.3.2

Description Calculate and extract remote sensing metrics for spatial analysis in the field of health. The package offers R users a quick and straightforward way to obtain areal or zonal statistics of key environmental indicators, covariates, and vector-borne disease data ideal for modeling infectious diseases within the framework of spatial epidemiology.

Maintainer Yomali Ferreyra <yomali.ferreyra@upch.pe>

URL https://github.com/harmonize-tools/land4health/, https://harmonize-tools.github.io/land4health/

License MIT + file LICENSE

Encoding UTF-8

Imports cli (>= 3.4.0), lifecycle, tidyr, dplyr, rgee, sf, httr2, ows4R, reticulate, rappdirs, tibble, terra

Suggests geojsonio, rmarkdown, knitr, testthat (>= 3.0.0), rstudioapi, geoidep, withr

Depends R (>= 4.1.0)

VignetteBuilder knitr

BugReports https://github.com/harmonize-tools/land4health/issues

Config/testthat/edition 3

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Antony Barja [aut, cph] (ORCID: <https://orcid.org/0000-0001-5921-2858>), Yomali Ferreyra [cre] (ORCID: <https://orcid.org/0000-0002-5184-9595>), Diego Villa [ctb], Gabriel Carrasco-Escobar [aut] (ORCID: <https://orcid.org/0000-0002-6945-0419>)

Repository CRAN

Date/Publication 2026-11-10 21:10:02 UTC

Contents

l4h_chirps 2
l4h_co_column 5
l4h_dengue 7
l4h_era5land 9
l4h_forest_loss 13
l4h_human_built 15
l4h_install 17
l4h_list_metrics 19
l4h_malaria 20
l4h_night_lights 22
l4h_packages 24
l4h_pm2_5 25
l4h_rural_access_index 27
l4h_sebal_modis 30
l4h_surface_temp 32
l4h_terra_climate 35
l4h_travel_time 38
l4h_urban_heat_index 40
l4h_urban_rural_area 42
l4h_use_python 44
l4h_vegetation 45

Index **49**

`l4h_chirps` *Extract CHIRPS v3 precipitation data from Google Earth Engine*

Description

Extracts **CHIRPS v3** precipitation estimates for a user-defined region and time range from the Earth Engine dataset **UCSB-CHC/CHIRPS/V3/DAILY_SAT** (IMERG-based) or **UCSB-CHC/CHIRPS/V3/DAILY_RNL** (ERA5-based). The function supports daily, monthly, and annual temporal aggregation. Monthly and annual values are computed as **sums** of daily precipitation (mm), which is the standard in hydrology and vector-borne disease modelling.

CHIRPS v3 (Climate Hazards Center InfraRed Precipitation with Stations version 3) is a quasi-global (60°S–60°N), high-resolution (0.05°) gridded rainfall dataset from 1981 to near-present, combining satellite thermal infrared estimates with in-situ station observations.



Usage

```
l4h_chirps(
  from,
  to,
  by = "month",
```

```
  product = "sat",
  region,
  scale = 5566,
  stat = "mean",
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

from	Character or Date. Start date ("YYYY-MM-DD").
to	Character or Date. End date ("YYYY-MM-DD").
by	Character. Temporal aggregation frequency. Options: • "daily" — daily precipitation (mm/day), • "month" — monthly accumulated precipitation (mm/month), • "annual" — annual accumulated precipitation (mm/year). Default "month".
product	Character. CHIRPS v3 daily product used for disaggregation. Options: • "sat" — IMERG-based daily (0.1° resolution, available from 2001), • "rnl" — ERA5-based daily (0.25° resolution, available from 1981). Default "sat".
region	Spatial object defining the region of interest. Accepts an sf, sfc, or SpatVector object.
scale	Numeric. Reducer scale in meters. Default 5566 (CHIRPS native pixel ~5,566 m).
stat	Character. Summary statistic per image per region. One of "mean", "median", "min", "max".
sf	Logical. If TRUE, returns an sf; if FALSE, returns a tibble. Default TRUE.
quiet	Logical. If TRUE, suppresses progress bars/messages. Default FALSE.
force	Logical. If TRUE, skips the representativity check. Default FALSE.
...	Additional arguments passed to the extraction backend.

Value

An sf or tibble with columns:

- date (Date — first day of the period),
- variable (character — "precipitation"),
- value (numeric — mm in the corresponding time unit), plus geometry if sf = TRUE, and any attributes from region.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

Funk, C., Peterson, P., Harrison, L. et al. (2026). The Climate Hazards Center Infrared Precipitation with Stations, Version 3. *Scientific Data*, 13, 718. doi:10.1038/s41597026070964

Examples

```
if (interactive()) {
library(land4health)
rgee::ee_Initialize()

# ROI simple (EPSG:4326)
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))), crs = 4326))

# 1. Monthly precipitation (mm/month) 2020, SAT product
out_monthly <- l4h_chirps(
  from    = "2020-01-01",
  to      = "2020-12-31",
  by      = "month",
  product = "sat",
  region  = region,
  stat    = "mean"
)
head(out_monthly)

# 2. Annual precipitation (mm/year) 2015-2020, RNL product
out_annual <- l4h_chirps(
  from    = "2015-01-01",
```

```
  to      = "2020-12-31",
  by      = "annual",
  product = "rnl",
  region  = region,
  stat    = "mean"
)
head(out_annual)

# 3. Daily precipitation (mm/day) for one month
out_daily <- l4h_chirps(
  from    = "2020-06-01",
  to      = "2020-06-30",
  by      = "daily",
  product = "sat",
  region  = region,
  stat    = "mean"
)
head(out_daily)
}
```

`l4h_co_column` *Extracts carbon monoxide (CO) concentration from Sentinel-5P TROPOMI*

Description

Retrieves the CO column number density (mol/m2) for a user-defined region and date range from the Sentinel-5P TROPOMI OFFLINE L3 CO dataset.



Usage

```
l4h_co_column(
  from,
  to,
  region,
  stat = "mean",
  scale = 1113,
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

from	Character. Start date in "YYYY-MM-DD" format (e.g., "2020-01-01").
to	Character. End date in "YYYY-MM-DD" format (e.g., "2020-12-31").
region	A spatial object (sf, sfc, or SpatVector) defining the region of interest.
stat	Character. Summary statistic to apply ("mean", "median", "max", etc.).
scale	Numeric. Nominal scale in meters. Default is 1113.
sf	Logical. Return result as sf? Default is TRUE.
quiet	Logical. Suppress progress messages? Default is FALSE.
force	Logical. Force extract without spatial check? Default is FALSE.
...	Arguments passed to the extraction backend.

Details

The function uses the Earth Engine dataset COPERNICUS/S5P/OFFL/L3_CO and selects only the "CO_column_number_density" band. Images are composited to daily means before extraction to avoid exceeding the 5000-band limit on toBands().

Value

An sf or tibble containing CO column density (mol/m2) by date and geometry.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

COPERNICUS/S5P/OFFL/L3_CO. Sentinel-5P Offline L3 Carbon Monoxide. European Union / ESA / Copernicus. https://developers.google.com/earth-engine/datasets/catalog/COPERNICUS_S5P_OFFL_L3_CO

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define region as a bounding box polygon
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326)
)

# Run CO column calculation
co_data <- l4h_co_column(
  from = "2022-01-01",
  to = "2022-12-31",
  region = region,
  stat = "mean"
)
head(co_data)
}
```

---

l4h_dengue *Extract dengue case data from OpenDengue*

Description

Downloads dengue case counts from the **OpenDengue Project**, a harmonized, open-access repository of dengue surveillance data from national ministries of health. The function supports national, spatial, and temporal extracts filtered by WHO region and country for a specified date range.



Usage

```
l4h_dengue(
  from,
  to,
  data_type = c("temporal", "spatial", "national"),
  region = NULL,
  country = "Peru",
  cache = TRUE,
```

```
  quiet = FALSE
)
```

Arguments

from	Character or Date. Start date ("YYYY-MM-DD").
to	Character or Date. End date ("YYYY-MM-DD").
data_type	Character. One of "national", "spatial", or "temporal". Default "temporal".
region	Character. WHO region code or full name (case-insensitive). Codes: "paho", "searo", "wpro", "afro", "emro", "euro". Full names: "Pan-American Region", "South-East Asia Region", "Western Pacific Region", "African Region", "Eastern Mediterranean Region", "European Region". Default "paho".
country	Character. Country name (case-insensitive), matched against adm_0_name. Default "Peru".
cache	Logical. If TRUE, caches the downloaded ZIP locally. Default TRUE.
quiet	Logical. If TRUE, suppresses progress messages. Default FALSE.

Value

A tibble with columns: date_start, date_end, cases, state, area, plus other fields from the source data.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

Source

Data from the OpenDengue Project.

References

Morales, I. et al. (2024). OpenDengue: Harmonized dengue surveillance data for Latin America.

See Also

l4h_malaria()

Examples

```
if (interactive()) {
  # National extract for Peru in 2019
  df_nat <- l4h_dengue(
    from = "2019-01-01",
    to = "2019-12-31",
    data_type = "national",
    region = "paho",
    country = "peru",
    cache = TRUE,
    quiet = TRUE
  )
  head(df_nat)

  # Spatial extract for Brazil
  df_spat <- l4h_dengue(
    from = "2021-01-01",
    to = "2021-12-31",
    data_type = "spatial",
    region = "Pan-American Region",
    country = "brazil",
    cache = TRUE,
    quiet = TRUE
  )
  head(df_spat)

  # Temporal extract for Argentina
  df_temp <- l4h_dengue(
    from = "2020-01-01",
    to = "2020-12-31",
    data_type = "temporal",
    region = "PAHO",
    country = "Argentina",
    cache = TRUE,
    quiet = TRUE
  )
  head(df_temp)
}
```

---

l4h_era5land *Extract ERA5-Land climate variables from Google Earth Engine*

Description

Extracts **ERA5-Land** climate variables for a user-defined region and time range from the Earth Engine datasets **ECMWF/ERA5_LAND/DAILY_AGGR** (`by = "daily"`) or **ECMWF/ERA5_LAND/MONTHLY_AGGR** (`by = "month"`), plus **ECMWF/ERA5_LAND/HOURLY** for the diurnal bands `"rh_day"`/`"rh_night"`. Each image is summarized over the region using a chosen statistic (e.g., mean/median), values are converted to conventional units (degrees Celsius, mm, volume fraction), and the function returns an `sf` or `tibble`.

ERA5-Land is the land-component replay of the ECMWF ERA5 climate reanalysis (Copernicus Climate Data Store), with global coverage from 1950 to near-present at ~9 km (0.1 degrees) resolution. It complements `l4h_chirps()` (finer precipitation) and `l4h_terra_climate()` (finer monthly climatology) by providing **daily** exposure windows and soil-moisture variables that are key for vector-borne disease modelling (temperature/precipitation lags).



Usage

```
l4h_era5land(
  from,
  to,
  by = "month",
  band = "t2m",
  region,
  scale = 11132,
  stat = "mean",
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

`from`	Character or Date. Start date (`"YYYY-MM-DD"`).
`to`	Character or Date. End date (`"YYYY-MM-DD"`).
`by`	Character. Temporal resolution. Options: • `"daily"` — daily values from `ECMWF/ERA5_LAND/DAILY_AGGR`, • `"month"` — monthly values from `ECMWF/ERA5_LAND/MONTHLY_AGGR`. Default `"month"`. Diurnal bands (`"rh_day"`, `"rh_night"`) always use the hourly collection (see Details), aggregated to the same periods.
`band`	Character vector. One or more ERA5-Land variables to extract. Supported codes: • `"t2m"` (2m air temperature, K -> degrees C), • `"d2m"` (2m dewpoint temperature, K -> degrees C), • `"rh"` (mean relative humidity, % — derived in Earth Engine from `temperature_2m` and `dewpoint_temperature_2m` via the Magnus formula, see Details),

- "rh_day" (daytime relative humidity, % — mean over 06:00-18:00 Peru time from ECMWF/ERA5_LAND/HOURLY, see Details),
- "rh_night" (nighttime relative humidity, % — mean over 18:00-06:00 Peru time from ECMWF/ERA5_LAND/HOURLY, see Details),
- "pr" (total precipitation, m -> mm),
- "soil" (volumetric soil water layer 1, 0-7 cm, m3/m3),
- "pet" (potential evaporation, m -> mm). Default "t2m".

region — Spatial object defining the region of interest. Accepts an sf, sfc, or SpatVector object.

scale — Numeric. Reducer scale in meters. Default 11132 (ERA5-Land native pixel ~11,132 m).

stat — Character. Summary statistic per image per region. One of "mean", "median", "min", "max".

sf — Logical. If TRUE, returns an sf; if FALSE, returns a tibble. Default TRUE.

quiet — Logical. If TRUE, suppresses progress bars/messages. Default FALSE.

force — Logical. If TRUE, skips the representativity check. Default FALSE.

... — Additional arguments passed to the extraction backend.

Details

Relative humidity (band = "rh") is derived server-side in Earth Engine from the bands the function already uses (temperature_2m and dewpoint_temperature_2m), following the Magnus formula for saturation vapour pressure (Wright 1997 coefficients, same as the EU operational product derived from ERA5-Land):

$$e(T) = 611.21 \times \exp\left(\frac{17.502 \times T}{240.97 + T}\right)$$

$$RH = 100 \times \frac{e(Td)}{e(T)}$$

with T (air temperature) and Td (dewpoint) in degrees C, converted from Kelvin inside Earth Engine (- 273.15). The ratio simplifies to 100 * exp(b*c*(Td - T) / ((c + T) * (c + Td))) with b = 17.502 and c = 240.97. Values are clamped to 0–100%.

Note the aggregation order: DAILY_AGGR/MONTHLY_AGGR already store time-averaged T/Td, so rh is the humidity of the averaged temperatures, not the average of instantaneous humidity. This is the standard practice (same as the EU ERA5-Land product). Below 0 °C the water-saturation form is kept (no ice correction), which is acceptable for vector-borne disease modelling but documented here as a limitation.

Day and night are defined in Peru time (PET, UTC-5, no daylight saving): day is 06:00-18:00 PET and night is 18:00-06:00 PET. Since ECMWF/ERA5_LAND/HOURLY timestamps are UTC, "rh_day" averages the UTC hours 11-22 and "rh_night" the UTC hours 23, 00-10. Each hourly image is converted to RH with the same Magnus formula above and then averaged (mean of hourly means with equal weights). Diurnal bands therefore cannot be combined with "rh" or native bands in one call: they come from different collections with different time bases.

Value

An `sf` or `tibble` with columns:

- `date` (Date — first day of the period),
- `variable` (character — band code, e.g. `"t2m"`, `"rh"`),
- `value` (numeric — degrees C, percent, mm, or volume fraction), plus geometry if `sf = TRUE`, and any attributes from `region`.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

Muñoz Sabater, J. (2019): ERA5-Land monthly averaged data from 1950 to present. Copernicus Climate Change Service (C3S) Climate Data Store (CDS). doi:10.24381/cds.68d2bb30

Bolton, D. (1980). The computation of equivalent potential temperature. *Monthly Weather Review*, 108, 1046-1053.

Alduchov, O. A., & Eskridge, R. E. (1996). Improved Magnus form approximation of saturation vapor pressure. *Journal of Applied Meteorology*, 35, 601-609.

Examples

```
if (interactive()) {
library(land4health)
rgee::ee_Initialize()

# ROI simple (EPSG:4326)
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))), crs = 4326))
```

```
# 1. Monthly 2m temperature (degrees C) 2020
out_monthly <- l4h_era5land(
  from   = "2020-01-01",
  to     = "2020-12-31",
  by     = "month",
  band   = "t2m",
  region = region,
  stat   = "mean"
)
head(out_monthly)

# 2. Daily relative humidity (%) + precipitation (mm), one month
out_daily <- l4h_era5land(
  from   = "2020-06-01",
  to     = "2020-06-30",
  by     = "daily",
  band   = c("rh", "pr"),
  region = region,
  stat   = "mean"
)
head(out_daily)

# 3. Daytime vs nighttime relative humidity (%), one week (Peru time)
out_diurnal <- l4h_era5land(
  from   = "2020-06-01",
  to     = "2020-06-07",
  by     = "daily",
  band   = c("rh_day", "rh_night"),
  region = region,
  stat   = "mean"
)
head(out_diurnal)
}
```

`l4h_forest_loss` *Extracts forest cover loss within a defined polygon*

Description

Calculates forest loss within a user-defined region for a specified year range. Forest loss is defined as a **stand-replacement disturbance**, or a change from forest to non-forest state.



Usage

```
l4h_forest_loss(from, to, region, sf = TRUE, quiet = FALSE, force = FALSE, ...)
```

Arguments

`from`	Character. Start date in `"YYYY-MM-DD"` format (only the year is used).
`to`	Character. End date in `"YYYY-MM-DD"` format (only the year is used).
`region`	A spatial object defining the region of interest. Accepts an `sf`, `sfc`, or `SpatVector` object (from the **terra** package).
`sf`	Logical. Return result as an `sf` object? Default is `TRUE`.
`quiet`	Logical. If `TRUE`, suppress the progress bar (default `FALSE`).
`force`	Logical. Force request extract.
`...`	arguments of `ee_extract` of `rgee` packages.

Details

Forest loss is derived from the Hansen Global Forest Change dataset. The `lossyear` band encodes the year of forest cover loss as follows:

- Values range from **1** to **n**, where 1 corresponds to the year **2001** and n to the year **2000 + n**.
- A value of **0** indicates **no forest loss** detected.

Value

A `sf` or `tibble` object with **forest loss per year in square kilometers**.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

Hansen, M. C., Potapov, P. V., Moore, R., Hancher, M., Turubanova, S. A., Tyukavina, A., ... & Townshend, J. R. G. (2013). *High-Resolution Global Maps of 21st-Century Forest Cover Change*. Science, 342(6160), 850–853. DOI: doi:10.1126/science.1244693

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define region as a bounding box polygon
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))

# Run forest loss calculation
result <- l4h_forest_loss(
  from = '2005-01-01',
  to = '2007-01-01',
  region = region)

head(result)
}
```

---

l4h_human_built *Extracts built-up surface area from GHSL Built-Up Surface dataset*

Description

Retrieves total built-up surface area (in m2 per 100m grid cell) from the GHSL Built-Up Surface dataset (GHS-BUILT-S R2023A), over a user-defined region and date range. The dataset is provided in 5-year epochs (1975–2030) at ~100m resolution.



Usage

```
l4h_human_built(
  from,
  to,
  region,
  scale = 100,
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
```

```
  ...
)
```

Arguments

`from`	Character. Start date in "YYYY-MM-DD" format (only the year is used).
`to`	Character. End date in "YYYY-MM-DD" format (only the year is used).
`region`	Spatial object (`sf`, `sfc`, or `SpatVector`) defining the region.
`scale`	Numeric. Resolution in meters (default = 100).
`sf`	Logical. If `TRUE`, returns an `sf`; if `FALSE`, returns a `tibble`. Default = `TRUE`.
`quiet`	Logical. If `TRUE`, suppresses progress output. Default = `FALSE`.
`force`	Logical. If `TRUE`, bypass representativity checks. Default = `FALSE`.
`...`	Arguments passed to `rgee::ee_extract()`.

Value

A `sf` or `tibble` with columns `date`, `variable`, and `built_surface_m2`.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

- Pesaresi, M. & Politis, P. (2023). GHS-BUILT-S R2023A: Red de superficie construida de GHS, derivada de la composición de Sentinel-2 y Landsat, multitemporal (1975–2030). European Commission, Joint Research Centre (JRC). doi:10.2905/9F06F36F4B1147ECABB0-4F8B7B1D72EA.
- Pesaresi, M., Schiavina, M., Politis, P., Freire, S., Krasnodebska, K., Uhl, J.H., Carioli, A., et al. (2024). Avances en la capa de asentamientos humanos globales a través de la evaluación conjunta de datos de observación de la Tierra y encuestas demográficas. *International Journal of Digital Earth*, 17(1). doi:10.1080/17538947.2024.2390454
- Dataset on Google Earth Engine: https://developers.google.com/earth-engine/datasets/catalog/JRC_GHSL_P2023A_GHS_BUILT_S

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define a bounding box region in Ucayali, Peru
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))

# Extract built-up surface area from 2000 to 2020
built_area <- l4h_human_built(
  from = "2000-01-01",
  to = "2020-12-31",
  region = region,
  scale = 100,
  stat = "sum"
)
head(built_area)

# Example using as tibble
built_tbl <- l4h_human_built(
  from = "1990-01-01",
  to = "2015-12-31",
  region = region,
  sf = FALSE,
  stat = "mean"
)
dplyr::glimpse(built_tbl)
}
```

`l4h_install` *Install Python dependencies for land4health package*

Description

Installs required Python packages (earthengine-api and numpy). By default uses **uv** when available (much faster downloads with a live progress bar in the console) and falls back to pip via `reticulate::py_install()` otherwise.



Usage

```
l4h_install(
  pip = TRUE,
  system = FALSE,
  force = FALSE,
  restart = TRUE,
  backend = c("auto", "uv", "pip"),
  ...
)
```

Arguments

pip	Logical. If TRUE (default), uses pip for installation. Set to FALSE if specifying a different installation method.
system	Logical. If TRUE, uses system pip directly via system() call.
force	Logical. If TRUE, forces reinstallation/upgrade of packages.
restart	Logical. If TRUE, automatically restarts R session after installation. Default TRUE.
backend	Character. Installation backend: "auto" (default) uses uv when it is installed and falls back to "pip" otherwise; "uv" requires uv (see https://docs.astral.sh/uv/); "pip" always uses reticulate::py_install(). Note: uv manages virtualenv environments only — an explicit method = "conda" always uses the "pip" backend.
...	Additional arguments passed to reticulate::py_install(), such as: • method: Installation method ("auto", "virtualenv", "conda") • envname: Environment name (default: "r-land4health")

Value

Invisibly returns NULL

Examples

```
# Installation is intentionally not run in examples because it modifies
# the Python environment and may be slow.

# See the installation vignette for instructions:
vignette("land4health-setup", package = "land4health")
```

`l4h_list_metrics` *List available metrics in* land4health

Description

Returns a tibble with the metadata of every metric shipped with **land4health**. You can optionally filter by thematic *category*, metric short name, or *provider*. If `open_in_browser = TRUE`, all matching URLs are opened in your default browser; for safety you **must** supply at least one filter, and—if more than five tabs would open—you are asked to confirm in interactive sessions.

Usage

```
l4h_list_metrics(
  category = NULL,
  metric = NULL,
  provider = NULL,
  open_in_browser = FALSE
)
```

Arguments

`category`	Optional single string. Filter by thematic category.
`metric`	Optional single string. Filter by metric short name.
`provider`	Optional single string. Filter by the `dataset` column.
`open_in_browser`	Logical; open the matching URLs in your browser? Defaults to `FALSE`.

Value

A tibble (returned *invisibly*). When `open_in_browser = FALSE` a compact preview (max 10 rows) is printed before the tibble is returned.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

Examples

```
## All examples keep `open_in_browser = FALSE` to avoid side-effects
## during automated checks.

## 1  Show the full inventory (truncated to 10 rows).
l4h_list_metrics()

## 2  Filter by category ("Human intervention").
l4h_list_metrics(category = "Human intervention")

## 3  Filter by provider ("WorldPop") and store the result.
worldpop_tbl <- l4h_list_metrics(provider = "WorldPop")
head(worldpop_tbl)
```

`l4h_malaria` *Extract malaria metrics from the Malaria Atlas Project GeoServer*

Description

Downloads modeled malaria raster surfaces from the **Malaria Atlas Project** (MAP) GeoServer via WCS 2.0.1, extracts zonal statistics for a user-defined region and time range, and returns an `sf` or `tibble`.

Available species: *Plasmodium falciparum* (`"pf"`) and *Plasmodium vivax* (`"pv"`). Available measures: `"incidence_rate"`, `"incidence_count"`, `"parasite_rate"`, `"mortality_rate"`, `"mortality_count"`. The function automatically selects the **latest release** available on the server for the requested species + measure combination.



Usage

```
l4h_malaria(
  from,
  to,
  species = "pf",
  measure = "incidence_rate",
  region,
  stat = "mean",
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

from	Integer. Start year.
to	Integer. End year. Must be >= from.
species	Character. One of "pf" (Plasmodium falciparum) or "pv" (Plasmodium vivax). Default "pf".
measure	Character. One of "incidence_rate", "incidence_count", "parasite_rate", "mortality_rate", "mortality_count". Default "incidence_rate".
region	Spatial object defining the region of interest. Accepts an sf, sfc, or SpatVector object.
stat	Character. Summary statistic per pixel per year. One of "mean", "median", "min", "max". Default "mean".
sf	Logical. If TRUE, returns an sf; if FALSE, returns a tibble. Default TRUE.
quiet	Logical. If TRUE, suppresses progress messages. Default FALSE.
force	Logical. If TRUE, skips the representativity check. Default FALSE.
...	Additional arguments (currently unused).

Details

The MAP GeoServer provides modeled global raster surfaces at ~5 km resolution. Rates are expressed per person (0-1 scale for parasite rate, per 1,000 for incidence), and counts are absolute numbers. The function connects to the GeoServer via ows4R::WCSClient, downloads the coverage clipped to the region bounding box, and uses terra::extract() for zonal statistics.

Value

An sf or tibble with columns:

- date (Date, first day of year),
- species (character, "pf" or "pv"),
- measure (character, the requested measure),
- value (numeric, in native units), plus geometry if sf = TRUE, and any attributes from region.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

Malaria Atlas Project. https://malariaatlas.org/

Examples

```
if (interactive()) {
library(land4health)

# ROI simple (EPSG:4326)
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))), crs = 4326))

# Pf incidence rate 2015-2020
out <- l4h_malaria(
  from    = 2015,
  to      = 2020,
  species = "pf",
  measure = "incidence_rate",
  region  = region
)
head(out)

# Pv parasite rate
out_pv <- l4h_malaria(
  from    = 2020,
  to      = 2022,
  species = "pv",
  measure = "parasite_rate",
  region  = region,
  sf      = FALSE
)
head(out_pv)
}
```

`l4h_night_lights` *Extracts global night-time lights using harmonized DMSP-OLS and VIIRS data*

Description

Retrieves annual night-time light radiance (average radiance, nanoWatt/sr/cm²) from the Harmonized Global Night Time Lights dataset for a user-defined region and time range. The dataset

harmonizes DMSP-OLS (1992-2013) with VIIRS-like data (2014-2021), ensuring consistent long-term time series at ~1km resolution.



Usage

```
l4h_night_lights(
  from,
  to,
  region,
  stat = "mean",
  scale = 1000,
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

`from`	Character. Start date in `"YYYY-MM-DD"` format (only the year is used).
`to`	Character. End date in `"YYYY-MM-DD"` format (only the year is used).
`region`	A spatial object (`sf`, `sfc`, or `SpatVector`) defining the region of interest.
`stat`	Character. Summary statistic to apply per year per region (e.g. `"mean"`, `"sum"`).
`scale`	Numeric. Nominal scale in meters (default `1000`).
`sf`	Logical. If `TRUE`, return as `sf`; if `FALSE`, return as `tibble`. Default `TRUE`.
`quiet`	Logical. If `TRUE`, suppress progress messages. Default `FALSE`.
`force`	Logical. If `TRUE`, skip representativity check. Default `FALSE`.
`...`	Additional arguments passed to `rgee::ee_extract()`.

Value

A `sf` or `tibble` with annual night-time light statistics per region and date.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define a bounding box region in Ucayali, Peru
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))

# Extract only DMSP-OLS data (1998-2010)
ntl_dmsp <- l4h_night_lights(
  from = "1998-01-01",
  to = "2010-12-31",
  region = region,
  stat = "mean"
)
head(ntl_dmsp)

# Extract only VIIRS data (2016-2021)
ntl_viirs <- l4h_night_lights(
  from = "2016-01-01",
  to = "2021-12-31",
  region = region,
  stat = "mean"
)
head(ntl_viirs)

# Extract both DMSP and VIIRS (2008-2020)
ntl_mixed <- l4h_night_lights(
  from = "2008-01-01",
  to = "2020-12-31",
  region = region,
  stat = "mean"
)
head(ntl_mixed)
}
```

---

l4h_packages *List all* land4health *packages*

Description

List all *land4health* packages

Usage

```
l4h_packages(include_self = TRUE)
```

Arguments

include_self	default TRUE. Includes the "land4health" package name in the resultant character vector.

Value

A character vector of package names included in the "land4health" meta-package.

Examples

```
l4h_packages()
```

---

l4h_pm2_5 *Extract Global PM2.5 (monthly) from Google Earth Engine*

Description

Extracts monthly **PM2.5** concentrations for a user-defined region and time range from the Earth Engine Community Catalog dataset **Global PM2.5 (V6GL02 CNN)**. Each monthly image is summarized over the region using a selected statistic (e.g., mean/median). The function returns either an `sf` or a `tibble`, with dates normalized to the **first day of each month**.



Usage

```
l4h_pm2_5(
  from,
  to,
  region,
  scale = 1000,
  stat = "mean",
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

from	Character or Date. Start date ("YYYY-MM-DD").
to	Character or Date. End date ("YYYY-MM-DD").
region	Spatial object defining the region of interest. Accepts an sf, sfc, or SpatVector object.
scale	Numeric. Reducer scale in meters. Default 1000. (Use a value close to the dataset's native grid; typical choices are a few km.)
stat	Character. Summary statistic per image per region. One of "mean", "median", "min", "max".
sf	Logical. If TRUE, returns an sf; if FALSE, returns a tibble. Default TRUE.
quiet	Logical. If TRUE, suppresses progress bars/messages. Default FALSE.
force	Logical. If TRUE, skips the representativity check (polygons smaller than 1 pixel are still extracted, only a warning is issued). Default FALSE.
...	Additional arguments passed to the extraction backend.

Details

This function queries the Global PM2.5 monthly product (V6GL02, CNN-based fusion) from the **GEE Community Catalog** and aggregates it over the provided region and dates. The dataset provides monthly surface PM2.5 concentrations (µg/m^3). Values are returned in native units (no extra scale factor is applied here).

Notes

- Dates are validated (YYYY-MM-DD) and constrained to the dataset range used in this package (default: 2000–2019).
- Output dates are normalized to the first day of each month found in the bands.
- The function expects a reasonable scale relative to the dataset resolution to avoid oversampling or excessive smoothing.

Value

An sf or tibble with columns:

- date (Date) — first day of the month,
- variable (character) — fixed as "pm2.5",
- value (numeric) — PM2.5 in **µg/m^3**, plus geometry if sf = TRUE, and any attributes from region.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

GEE Community Catalog – Global PM2.5 (V6GL02 CNN). https://gee-community-catalog.org/projects/global_pm25/

Examples

```
if (interactive()) {
library(land4health)
rgee::ee_Initialize()

# ROI simple (EPSG:4326)
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))), crs = 4326))

# Monthly PM2.5 (ug/m^3) for 2010, spatial mean
out_pm <- l4h_pm2_5(
  from   = "2010-01-01",
  to     = "2010-12-31",
  band   = "b1",        # ignored (single band)
  region = region,
  stat   = "mean",
  scale  = 3000
)
head(out_pm)
}
```

`l4h_rural_access_index`

Compute Rural Access Index (RAI)

Description

Calculates the Rural Access Index (RAI) for a given region using datasets from the GEE Community Catalog. The RAI represents the proportion of the rural population living within 2 km of an all-season road, aligning with SDG indicator 9.1.1.



Usage

```
l4h_rural_access_index(
  region,
  weighted = FALSE,
  fun = NULL,
  sf = FALSE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

`region`	A spatial object defining the region of interest. Can be an `sf`, `sfc` object, or a `SpatVector` (from the **terra** package).
`weighted`	Logical. If `TRUE`, computes a population-weighted RAI (i.e., rural population with access divided by total rural population). If `FALSE`, computes an area-based RAI (i.e., total pixel area with access divided by total rural area). Default is `FALSE`.
`fun`	Character. Summary function to apply to the population raster when `weighted = TRUE`. Common values include `"mean"`, `"sum"`, etc. Ignored when `weighted = FALSE`. Default is `"mean"`.
`sf`	Logical. If `TRUE`, returns the result as an `sf` object. If `FALSE`, returns an Earth Engine object. Default is `FALSE`.
`quiet`	Logical. If TRUE, suppress the progress bar (default FALSE).
`force`	Logical. If `TRUE`, skips the representativity check and forces the extraction. Default is `FALSE`.
`...`	arguments of `ee_extract` of `rgee` packages.

Details

This function uses the following datasets from the GEE Community Catalog:

- `projects/sat-io/open-datasets/RAI/ruralpopaccess/` – raster of rural population with access to all-season roads
- `projects/sat-io/open-datasets/RAI/inaccessibilityindex/` – binary raster indicating access areas (1 = access, 0 = no access)

When `weighted = TRUE`, the RAI is calculated as the sum (or chosen summary via `fun`) of the accessible rural population divided by the total rural population within the specified region.

When `weighted = FALSE`, the RAI is calculated as the ratio of pixel areas: the total area (in in km^2) with access divided by the total rural area.

The `fun` parameter only applies when `weighted = TRUE`. It will be ignored otherwise.

Value

A spatial object containing the computed RAI value for the region in an `sf` or `tibble` object.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

GEE Community Catalog: https://gee-community-catalog.org/projects/rai/

Frontiers in Remote Sensing (2024): doi:10.3389/frsen.2024.1375476

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define a bounding box region in Ucayali, Peru
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))
```

```
# Population-weighted RAI
rai_w <- l4h_rural_access_index(
    region = region,
    weighted = TRUE,
    fun = "sum",
    sf = TRUE)
head(rai_w)

# Area-based RAI
rai <- l4h_rural_access_index(
    region = region,
    weighted = FALSE,
    sf = TRUE)
head(rai)
}
```

`l4h_sebal_modis` *Download and process evapotranspiration data*

Description

This function accesses the geeSEBAL-MODIS collection published by the ET-Brasil project, extracts the `etp` band (daily evapotranspiration in mm/day), and allows temporal aggregation by 8-day images or monthly or yearly composites period. Optionally, results can be returned as `sf/tibble` objects in R.



Usage

```
l4h_sebal_modis(
  from,
  to,
  by = "8 days",
  region,
  fun = "mean",
  sf = TRUE,
  force = FALSE,
  quiet = FALSE,
  ...
)
```

Arguments

`from`	Start date in `"YYYY-MM-DD"` format.
`to`	End date in `"YYYY-MM-DD"` format.

by	Temporal aggregation frequency. Options: "8 days" (original 8-day composites), "month" (monthly average or sum), or "annual" (annual avergae or sumperiod).
region	A spatial object defining the region of interest. Accepts sf, SpatVector, or ee$FeatureCollection objects.
fun	Aggregation function when by = "month" or "total". Valid values are "mean" or "sum".
sf	Logical. Return result as an sf object? Default is TRUE.
force	Logical. If TRUE, forces download even if a local file already exists.
quiet	Logical. If TRUE, suppress the progress bar (default FALSE).
...	arguments of ee_extract of rgee packages.

Value

A sf or tibble object with etp values.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

- Comini,B., Ruhoff,A., Laipelt,L., Fleischmann,A., Huntington,J., Morton,C., Melton,F., Erickson,T., Roberti,D., Souza,V., Biudes,M., Machado,N., Santos,C. & Cosio,E. (2023). *geeSEBAL-MODIS: Continental-scale evapotranspiration based on the surface energy balance for South America.* Preprint. DOI: 10.13140/RG.2.2.17579.11041
- geeSEBAL-MODIS v0-02 dataset. Licensed under the *Creative Commons Attribution 4.0 International (CC-BY-4.0)* license.

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()
```

```
# Define a bounding box region in Ucayali, Peru
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))

# 1. Eight-day composites (8 days)
# 2020-01-01 → 2020-12-31, reducer = "mean"
sebal_8d <- l4h_sebal_modis(
  from   = "2020-01-01",
  to     = "2020-12-31",
  region = region
)

# 2. Monthly means
# Same period, but aggregated to calendar months
sebal_month <- l4h_sebal_modis(
  from   = "2020-01-01",
  to     = "2020-12-31",
  by     = "month",
  region = region
)

# 3. Annual evapotranspiration
# 2015 → 2022, one value per year
sebal_annual <- l4h_sebal_modis(
  from   = "2015-01-01",
  to     = "2022-12-31",
  by     = "annual",
  fun    = "sum",
  region = region,
  sf     = FALSE
)

}
```

`l4h_surface_temp` *Extracts Land Surface Temperature (LST) from MODIS MOD11A1*

Description

Extracts daytime or nighttime Land Surface Temperature (LST) for a user-defined region and time range using the MODIS MOD11A1.061 product. The function supports summarizing the temperature data over each date (or each month) using a selected statistic (e.g., mean or median).

lifecycle experimental

Usage

```
l4h_surface_temp(
  from,
  to,
  region,
  band = "day",
  level = "strict",
  by = "day",
  scale = 1000,
  stat = "mean",
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

from	Character or Date. Start date of the analysis (e.g., "2020-01-01").
to	Character or Date. End date of the analysis (e.g., "2020-12-31").
region	A spatial object defining the region of interest. Accepts an sf, sfc, or SpatVector object.
band	Character. LST type to extract: "day" (LST_Day_1km) or "night" (LST_Night_1km). Default is "day".
level	Character. Quality filter level to apply to MODIS LST pixels. Use "strict" to retain only high-quality observations (QA bits 0-1 equal to 00), or "moderate" to allow both high and acceptable quality (QA bits 0-1 equal to 00 or 01). Default is "moderate".
by	Character. Temporal resolution of the output. One of "day" (default, one value per available daily image) or "month". When "month", cloud-masked daily images within each calendar month are combined server-side with the reducer selected in stat (mean, median, min or max) *before* the spatial extraction, so masked (cloudy) pixels do not count as zeros or missing days — they are simply excluded from that month's reducer.
scale	Numeric. Spatial resolution in meters. Default is 1000 (native resolution).
stat	Character. Summary statistic to apply. One of "mean", "median", "min", "max". Used as the spatial reducer passed to ee_extract() for every by value, and additionally as the temporal reducer across days within a month when by = "month".
sf	Logical. If TRUE, returns an sf object; if FALSE, returns a tibble. Default is TRUE.
quiet	Logical. If TRUE, suppresses progress bars and messages. Default is FALSE.

force	Logical. If TRUE, skips the representativity check (polygons smaller than 1 pixel are still extracted, only a warning is issued). Default is FALSE.
...	Additional arguments passed to rgee::ee_extract().

Details

The MODIS MOD11A1.061 product provides daily Land Surface Temperature and quality information. This function filters out low-quality or cloud-contaminated pixels based on the QC_Day or QC_Night band.

When by = "month", for each calendar month in [from, to] the function:

1. Filters the daily collection to that month.
2. Applies the same quality mask used for daily extraction to every image.
3. Reduces the masked images to a single monthly image using the reducer implied by stat.

Because masking happens before reducing, a cloudy day never drags the monthly value down or up — it simply does not contribute a pixel to that month's calculation.

LST values are originally stored as Kelvin multiplied by 0.02. This function automatically converts them to degrees Celsius using the formula: LST = (value x 0.02) - 273.15.

Value

A sf or tibble object with LST values (in degrees Celsius) extracted from MODIS MOD11A1, at daily or monthly resolution depending on by.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

References

Wan, Z., Hook, S., & Hulley, G. (2015). MOD11A1 MODIS/Terra Land Surface Temperature and Emissivity Daily L3 Global 1km SIN Grid V006 (Version 6.1). NASA EOSDIS Land Processes DAAC. doi:10.5067/MODIS/MOD11A1.061

MODIS MOD11A1.061 - Google Earth Engine Dataset Catalog. https://developers.google.com/earth-engine/datasets/catalog/MODIS_061_MOD11A1

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
```

```
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))

# Daily (unchanged default behaviour)
lst_day <- l4h_surface_temp(
  from = "2020-01-01", to = "2020-12-31",
  region = region, band = "day", stat = "mean")

# Monthly
lst_month <- l4h_surface_temp(
  from = "2020-01-01", to = "2020-12-31",
  region = region, band = "day", stat = "mean", by = "month")

head(lst_month)
}
```

`l4h_terra_climate` *Extract TerraClimate variables (monthly) from Google Earth Engine*

Description

Extracts one or more **TerraClimate** variables for a user-defined region and time range from the Earth Engine dataset **IDAHO_EPSCOR/TERRACLIMATE**. The function summarizes each monthly image over the region using a chosen statistic (e.g., mean/median), applies the appropriate **scale factors** to return values in native units, and returns an `sf` or `tibble`.



Usage

```
l4h_terra_climate(
  from,
  to,
  band,
  region,
  scale = 1000,
  stat = "mean",
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

from	Character or Date. Start date ("YYYY-MM-DD").
to	Character or Date. End date ("YYYY-MM-DD").
band	Character vector. One or more TerraClimate variables to extract. Supported codes: "aet", "def", "pdsi", "pet", "pr", "ro", "soil", "srad", "swe", "tmmn", "tmmx", "vap", "vpd", "vs". Scale factors and units (applied automatically): • aet (mm, ×0.1), def (mm, ×0.1), pdsi (unitless, ×0.01), • pet (mm, ×0.1), pr (mm, ×1), ro (mm, ×1), soil (mm, ×0.1), • srad (W/m², ×0.1), swe (mm, ×1), • tmmn (°C, ×0.1), tmmx (°C, ×0.1), • vap (kPa, ×0.001), vpd (kPa, ×0.01), vs (m/s, ×0.01).
region	Spatial object defining the region of interest. Accepts an sf, sfc, or SpatVector object.
scale	Numeric. Reducer scale in meters. Default 1000. (TerraClimate pixel approx 4638 m; a range of 4500-5000 m is typically appropriate.)
stat	Character. Summary statistic per image per region. One of "mean", "median", "min", "max". Passed internally to the extractor.
sf	Logical. If TRUE, returns an sf; if FALSE, returns a tibble. Default TRUE.
quiet	Logical. If TRUE, suppresses progress bars/messages. Default FALSE.
force	Logical. If TRUE, skips the representativity check (polygons smaller than 1 pixel are still extracted, only a warning is issued). Default FALSE.
...	Additional arguments passed to the extraction backend.

Value

An sf or tibble with columns:

- date (Date, first day of the month),
- variable (character, TerraClimate code),
- value (numeric, in native units with scale factors applied), plus geometry if sf = TRUE, and any attributes from region.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook

- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

Abatzoglou, J. T., Dobrowski, S. Z., Parks, S. A., & Hegewisch, K. C. (2018). TerraClimate, a high-resolution global dataset of monthly climate and climatic water balance from 1958–2015. *Scientific Data*, 5, 170191. doi:10.1038/sdata.2017.191

Examples

```
if (interactive()) {
library(land4health)
rgee::ee_Initialize()

# ROI simple (EPSG:4326)
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))), crs = 4326))

# Monthly precipitation (mm) 2020, spatial mean
out_pr <- l4h_terra_climate(
  from = "2020-01-01",
  to   = "2020-12-31",
  band = "pr",
  region = region,
  stat = "mean",
  scale = 5000
)
head(out_pr)

# Multiple variables: Tmax (C) + VPD (kPa)
out_multi <- l4h_terra_climate(
  from = "2019-01-01",
  to   = "2019-12-31",
  band = c("tmmx","vpd"),
  region = region,
  stat = "median",
  scale = 5000
)
}
```

`l4h_travel_time` *Travel Time to Healthcare or Cities (Oxford Dataset)*

Description

Retrieves the travel time raster (in minutes) to the nearest healthcare facility or populated city, based on the Oxford Global Map of Accessibility datasets.



Usage

```
l4h_travel_time(
  region,
  destination = "cities",
  transport_mode = "all",
  fun = "mean",
  sf = FALSE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

`region`	A spatial object defining the region of interest. Can be an `sf`, `sfc` object, or a `SpatVector` (from the **terra** package).
`destination`	Character. Target destination for travel time. Use `"healthcare"` (default) for travel time to the nearest healthcare facility, or `"cities"` for travel time to the nearest populated urban center.
`transport_mode`	Character. Mode of transportation. Use `"all"` (default) for general travel time (mixed modes), or `"walking_only"` for walking-only accessibility (**only valid when** `destination = "healthcare"`).
`fun`	Character. Summary function to apply. Values include `"mean"`, `"sum"`,`"median"` , etc. Default is `"mean"`.
`sf`	Logical. If `TRUE`, returns the result as an `sf` object. If `FALSE`, returns an Earth Engine object. Default is `FALSE`.
`quiet`	Logical. If TRUE, suppress the progress bar (default FALSE).
`force`	Logical. If `TRUE`, skips the internal representativity check of the input region. Defaults to `FALSE`.
`...`	arguments of `ee_extract` of `rgee` packages.

Value

A spatial object containing the computed RAI value for the region in an `sf` or `tibble` object.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

- Weiss, D.J. et al. (2018). *A global map of travel time to cities to assess inequalities in accessibility in 2015.* Nature, 553(7688), 333–336. DOI: 10.1038/nature25181
- Weiss, D.J. et al. (2020). *Global maps of travel time to healthcare facilities.* Nature Medicine, 26, 1835–1838. DOI: 10.1038/s41591-020-1059-1

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define a bounding-box region in Ucayali, Peru
region <- st_as_sf(
  st_sfc(
    st_polygon(list(matrix(
      c(
        -74.1, -4.4,
        -74.1, -3.7,
        -73.2, -3.7,
        -73.2, -4.4,
        -74.1, -4.4
      ), ncol = 2, byrow = TRUE
    )))
  ),
  crs = 4326
)

# Travel time to nearest healthcare facility (all modes)
result_hosp_all <- l4h_travel_time(region = region)
head(result_hosp_all)

# Travel time to nearest healthcare facility (walking only)
result_hosp_walk <- l4h_travel_time(
```

```
  region        = region,
  destination   = "healthcare",
  transport_mode = "walking_only")

head(result_hosp_walk)

# Mean travel time to nearest cities (mixed modes)
result_city_mean <- l4h_travel_time(
  region      = region,
  destination = "cities",
  fun         = "mean")

head(result_city_mean)

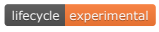
# Sum of travel time to nearest cities
result_city_sum <- l4h_travel_time(
  region      = region,
  destination = "cities",
  fun         = "sum")

head(result_city_sum)
}
```

`l4h_urban_heat_index` *Calculates the Surface Urban Heat Island (SUHI) index using MODIS LST and GHS-SMOD*

Description

Computes the SUHI (Surface Urban Heat Island) index as the difference between the mean land surface temperature (LST) in urban and rural areas for each date in a user-defined region and time range.



Usage

```
l4h_urban_heat_index(
  from,
  to,
  region,
  band = "day",
  level = "strict",
  stat = "max",
  scale = 1000,
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
```

```
  ...
)
```

Arguments

from	Character or Date. Start date (format: "YYYY-MM-DD").
to	Character or Date. End date (format: "YYYY-MM-DD").
region	Spatial object (sf, sfc, or SpatVector) defining the region.
band	Character. "day" or "night" LST from MODIS. Default is "day".
level	Character. "strict" or "moderate" quality filter for MODIS. Default is "strict".
stat	Character. Aggregation statistic, e.g. "mean" or "median". Default is "mean".
scale	Numeric. Resolution in meters. Default is 1000.
sf	Logical. If TRUE, returns an sf; if FALSE, returns a tibble. Default is TRUE.
quiet	Logical. If TRUE, suppress messages. Default is FALSE.
force	Logical. If TRUE, skip representativity check. Default is FALSE.
...	Extra arguments passed to ee_extract().

Value

A tibble or sf object with columns: date, variable = "SUHI", and value (°C).

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define a bounding box region (Ucayali, Peru)
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
```

```
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))

# Calculate SUHI using daytime LST (mean temperature difference)
suhi_day <- l4h_urban_heat_index(
  from = "2020-01-01",
  to = "2020-12-31",
  region = region,
  band = "day",
  stat = "mean"
)
head(suhi_day)

# Calculate SUHI using nighttime LST (max difference)
suhi_night <- l4h_urban_heat_index(
  from = "2020-01-01",
  to = "2020-12-31",
  region = region,
  band = "night",
  stat = "max"
)
head(suhi_night)
}
```

`l4h_urban_rural_area` *Extracts surface areas by urban and rural categories from GHS-SMOD*

Description

Calculates the surface area (in km2) of urban, rural, or all settlement classes every 5 years between 1985 and 2030 using the GHS-SMOD R2023A dataset. This product applies the Degree of Urbanization methodology (Stage I) to the GHS-POP R2023A and GHS-BUILT-S R2023A layers. The function summarizes areas by category and year over the specified region.



Usage

```
l4h_urban_rural_area(
  region,
  category = "all",
  scale = 1000,
```

```
  sf = TRUE,
  quiet = FALSE,
  force = FALSE,
  ...
)
```

Arguments

region	An sf object defining the region of interest.
category	Character. Settlement category to extract: "urban", "rural", or "all".
scale	Numeric. Spatial resolution (in meters) to use for area calculation (e.g., 30).
sf	Logical. If TRUE, returns an sf object. Default is TRUE.
quiet	Logical. If TRUE, suppresses progress messages. Default is FALSE.
force	Logical. If TRUE, forces the extraction request even if cached results exist.
...	Additional arguments passed to ee_extract() from the rgee package.

Value

A tibble with estimated settlement area (in km2) by year and category.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

References

- European Commission, Joint Research Centre (JRC). GHS Settlement Grid R2023A (1975–2030). Available at: https://data.jrc.ec.europa.eu/dataset/a0df7a6f-49de-46ea-9bde-563437a6e2ba#dataaccess

Examples

```
if (interactive()) {
library(land4health)
ee_Initialize()

# Define region as a bounding box (Ucayali, Peru)
region <- st_as_sf(st_sfc(
  st_polygon(list(matrix(c(
    -74.1, -4.4,
    -74.1, -3.7,
    -73.2, -3.7,
    -73.2, -4.4,
    -74.1, -4.4
  ), ncol = 2, byrow = TRUE))),
  crs = 4326
))

# Extract surface area of urban category (in km2)
urban_area <- l4h_urban_rural_area(
  category = "urban",
  region = region)

head(urban_area)

# Extract surface area of rural category (in km2)
rural_area <- l4h_urban_rural_area(
  category = "rural",
  region = region)

head(rural_area)

# Extract total surface area (urban + rural) (in km2)
all_area <- l4h_urban_rural_area(
  category = "all",
  region = region)

head(all_area)
}
```

`l4h_use_python` *Configure Python environment for land4health*

Description

Sets up the Python environment automatically based on installation.

Usage

```
l4h_use_python(envname = NULL, method = NULL, quiet = FALSE)
```

Arguments

envname	Character. Name of the Python environment. If NULL, uses saved config.
method	Character. Method to use ("auto", "virtualenv", "conda"). If NULL, uses saved config.
quiet	Logical. Suppress messages? Default FALSE.

Value

Invisibly returns NULL

Examples

```
if (interactive()) {
l4h_use_python()
l4h_use_python("r-land4health", "virtualenv")
}
```

---

l4h_vegetation *Extract vegetation indices from MODIS MOD13A1*

Description

Computes monthly or annual areal statistics of vegetation indices (NDVI, EVI, or SAVI) from MODIS MOD13A1 (500 m, 16-day composite) for a given spatial region, applying quality filtering via the DetailedQA bitmask.



Usage

```
l4h_vegetation(
  region,
  from,
  to,
  band = c("NDVI", "EVI", "SAVI"),
  by = c("month", "year"),
  fun = c("mean", "max", "min", "median", "sum", "sd", "first"),
  scale = 500,
  sf = FALSE,
  quiet = FALSE,
  force = FALSE
)
```

Arguments

region	An sf object (polygon or multipolygon). Must be in a geographic CRS (or will be reprojected to WGS84 internally).
from	Character. Start date in "YYYY-MM-DD" format. Valid range: "2000-02-18" onwards.
to	Character. End date in "YYYY-MM-DD" format.
band	Character. Vegetation index to extract. One of "NDVI", "EVI", or "SAVI". Default: "NDVI".
by	Character. Temporal aggregation unit. One of "month" (default) or "year".
fun	Character. Zonal statistic to compute over the region. One of "mean", "max", "min", "median", "sum", "sd", "first". Default: "mean".
scale	Numeric. Nominal scale in metres for the GEE projection. Default: 500 (native MOD13A1 resolution).
sf	Logical. If TRUE, returns an sf object with geometries attached. Default: FALSE.
quiet	Logical. If TRUE, suppresses the progress bar. Default: FALSE.
force	Logical. If TRUE, skips the representativity check (polygons smaller than 1 pixel are still extracted, only a warning is issued). Default: FALSE.

Details

Temporal aggregation:

MODIS MOD13A1 produces one composite every **16 days**. This function aggregates those composites into a coarser temporal unit:

- by = "month": All 16-day images within each calendar month are reduced to a single image using max() (maximum value composite), yielding one value per region per month.
- by = "year": All 16-day images within each calendar year are reduced to a single image using max(), yielding one value per region per year.

The fun argument controls the **spatial** (zonal) statistic applied over each region polygon, and is independent of the temporal aggregation.

Quality filtering:

Applied through the DetailedQA bitmask of MODIS/061/MOD13A1:

- Bits 0-1: VI quality (value 2 = not produced/cloudy, excluded).
- Bit 14: Adjacent cloud detected (excluded).
- Bit 15: Possible shadow (excluded).

Scale factors:

- NDVI and EVI: multiplied by 0.0001.
- SAVI: computed on-the-fly from surface reflectance bands sur_refl_b01 (red) and sur_refl_b02 (NIR), L = 0.5.

Value

A tibble (or `sf` tibble if `sf = TRUE`) in long format:

`<id_cols>` Original attribute columns from `region`.

`date` `Date` object. First day of each month (`by = "month"`) or first day of each year (`by = "year"`).

`variable` Name of the vegetation index (e.g. `"NDVI"`).

`value` Computed zonal statistic for that region and period.

Credits



Pioneering geospatial health analytics and open-science tools. Developed by the Innovalab Team. For more information, send an email to <imt.innovlab@oficinas-upch.pe>.

Follow us on:

- Innovalab Linkedin
- Innovalab X
- Innovalab facebook
- Innovalab instagram
- Innovalab tiktok
- Innovalab Podcast

Examples

```
if (interactive()) {
library(land4health)
library(geoidep)

rgee::ee_Initialize(quiet = TRUE)

provinces <- get_provinces(show_progress = FALSE) |>
  subset(nombdep == "LORETO")

# Monthly mean NDVI
result_monthly <- provinces |>
  l4h_vegetation(
    from = "2010-01-01",
    to   = "2012-12-31",
    band = "NDVI",
    by   = "month",
    fun  = "mean",
    sf   = TRUE
  )

 head(result_monthly)

# Annual mean NDVI
```

```
result_annual <- provinces |>
  l4h_vegetation(
    from = "2010-01-01",
    to   = "2020-12-31",
    band = "NDVI",
    by   = "year",
    fun  = "mean",
    sf   = TRUE
  )

 str(result_annual)

}
```

Index

l4h_chirps, 2
l4h_co_column, 5
l4h_dengue, 7
l4h_era5land, 9
l4h_forest_loss, 13
l4h_human_built, 15
l4h_install, 17
l4h_list_metrics, 19
l4h_malaria, 20
l4h_malaria(), 9
l4h_night_lights, 22
l4h_packages, 24
l4h_pm2_5, 25
l4h_rural_access_index, 27
l4h_sebal_modis, 30
l4h_surface_temp, 32
l4h_terra_climate, 35
l4h_travel_time, 38
l4h_urban_heat_index, 40
l4h_urban_rural_area, 42
l4h_use_python, 44
l4h_vegetation, 45