Package ‘modisfast’

November 12, 2024

Title Fast and Efficient Access to MODIS Earth Observation Data

Version 1.0.0

Description Programmatic interface to several NASA Earth Observation 'OPeNDAP' servers (Open-source Project for a Network Data Access Protocol) (<https://www.opendap.org/>). Allows for easy downloads of MODIS subsets, as well as other Earth Observation datacubes, in a time-saving and efficient way : by sampling it at the very downloading phase (spatially, temporally and dimensionally).

License GPL (>= 3)

URL https://github.com/ptaconet/modisfast

BugReports https://github.com/ptaconet/modisfast/issues

Depends R (>= 2.10)

Imports curl, dplyr, httr, lubridate, magrittr, parallel, purrr, rvest, sf, stats, stringr, terra, xml2, cli

Suggests ggplot2, knitr, mapview, rmarkdown, spelling, testthat, appeears

Encoding UTF-8

LazyData true

RoxygenNote 7.3.1

Language en-US

Config/testthat/start-first mf_login, mf_list_collections, mf_list_variables, mf_get_url, mf_download_data, mf_import_data

NeedsCompilation no

Author Paul Taconet [aut, cre, cph] (<https://orcid.org/0000-0001-7429-7204>), Nicolas Moiroux [fnd] (<https://orcid.org/0000-0001-6755-6167>), French National Research Institute for Sustainable Development, IRD [fnd]

Maintainer Paul Taconet <paul.taconet@gmail.com>

Repository CRAN

Date/Publication 2024-11-12 15:00:02 UTC

Contents

## `entomological_data` *Example dataset containing abundances of mosquitoes vectors of malaria. Used in article 'use_case'.*

Description

Example dataset containing abundances of mosquitoes vectors of malaria. Used in article 'use_case'.

Usage

```
entomological_data
```

Format

## 'entomological_data' A data frame with 232 rows and 6 columns:

mission number of the entomological survey

date date of the survey

village 3-digit code for the village of the survey

X, Y longitude and latitude of the center of the village

n number of mosquitoes collected

Source

<https://doi.org/10.15468/v8fvyn>

mf_download_data *Download several datasets given their URLs and destination path*

Description

This function enables to download datasets. In a data import workflow, this function is typically used after a call to the mf_get_url function. The output value of mf_get_url can be used as input of parameter `df_to_dl` of mf_download_data.

The download can the parallelized.

Usage

```
mf_download_data(
  df_to_dl,
  path = tempfile("modisfast_"),
  parallel = FALSE,
  num_workers = parallel::detectCores() - 1,
  credentials = NULL,
  verbose = "inform",
  min_filesize = 5000
)
```

Arguments

`df_to_dl`	data.frame. Urls and destination files of dataset to download. Typically output of mf_get_url. See Details for the structure
`path`	string. Target folder for the data to download. Default : temporary folder.
`parallel`	boolean. Parallelize the download ? Default to FALSE
`num_workers`	integer. Number of workers in case of parallel download. Default to number of workers available in the machine minus one.
`credentials`	vector string of length 2 with username and password. optional if the function mf_login was previously executed.
`verbose`	string. Verbose mode ("quiet", "inform", or "debug"). Default "inform".
`min_filesize`	integer. Minimum file size expected (in bites) for one file downloaded. If files downloaded are less that this value, the files will be downloaded again. Default 5000.

Details

Parameter `df_to_dl` must be a data.frame with the following minimal structure :

id_roi An id for the ROI (character string)

collection Collection (character string)

name

url URL of the file to download (character string)

Value

a data.frame with the same structure of the input data.frame `df_to_dl` + columns providing details of the data downloaded. The additional columns are :

fileDl Booloean (dataset downloaded or failure)

dlStatus Download status : 1 = download ok ; 2 = download error ; 3 = dataset was already existing in destination file

fileSize File size on disk (in bites)

Examples

```
## Not run:

### Login to EOSDIS Earthdata with your username and password
log <- mf_login(credentials = c("earthdata_un", "earthdata_pw"))

### Set-up parameters of interest
coll <- "MOD11A1.061"

bands <- c("LST_Day_1km", "LST_Night_1km")

time_range <- as.Date(c("2017-01-01", "2017-01-30"))

roi <- sf::st_as_sf(
  data.frame(
    id = "roi_test",
    geom = "POLYGON ((-5.82 9.54, -5.42 9.55, -5.41 8.84, -5.81 8.84, -5.82 9.54))"
  ),
  wkt = "geom", crs = 4326
)

### Get the URLs of the data
(urls_mod11a1 <- mf_get_url(
  collection = coll,
  variables = bands,
  roi = roi,
  time_range = time_range
))

### Download the data
res_dl <- mf_download_data(urls_mod11a1)

### Import the data as terra::SpatRast
modis_ts <- mf_import_data(dirname(res_dl$destfile[1]), collection = coll)

### Plot the data
terra::plot(modis_ts)

## End(Not run)
```

`mf_get_opt_param` *Precompute the parameter* `opt_param` *of the function mf_get_url*

Description

Precompute the parameter `opt_param` to further provide as input of the mf_get_url function. Useful to speed-up the overall processing time.

Usage

```
mf_get_opt_param(collection, roi, credentials = NULL, verbose = "inform")
```

Arguments

`collection`	string. mandatory. Collection of interest (see details of mf_get_url).
`roi`	object of class `sf`. mandatory. Area of region of interest. Must be a Simple feature collection with geometry type POLYGON, composed of one or several rows (i.e. one or several ROIs), and with at least two columns: 'id' (an identifier for the roi) and 'geom' (the geometry).
`credentials`	vector string of length 2 with username and password. optional if the function mf_login was previously executed.
`verbose`	string. Verbose mode ("quiet", "inform", or "debug"). Default "inform".

Details

When it is needed to loop the function mf_get_url over several time frames, it is advised to previously run the function `mf_get_opt_param` and provide the output as input `opt_param` parameter of the mf_get_url function. This will save much time, as internal parameters will be calculated only once.

Value

a list with the following named objects :

roiSpatialIndexBound OPeNDAP indices for the spatial coordinates of the bounding box of the ROI (minLat, maxLat, minLon, maxLon)

availableVariables Variables available for the collection of interest

roiSpatialBound The spatial coordinates of the bounding box of the ROI expressed in the CRS of the collection

OpenDAPXVector The X (longitude) vector

OpenDAPYVector The Y (longitude) vector

OpenDAPtimeVector The time vector, or NULL if the collection does not have a time vector

modis_tile The MODIS tile(s) number(s) for the ROI or NULL if the collection is not MODIS

Examples

```
## Not run:

# Login to Earthdata

log <- mf_login(credentials = c("earthdata_un", "earthdata_pw"))

# Get the optional parameters for the collection MOD11A1.061 and the following roi :
roi <- sf::st_as_sf(
  data.frame(
    id = "roi_test",
    geom = "POLYGON ((-5.82 9.54, -5.42 9.55, -5.41 8.84, -5.81 8.84, -5.82 9.54))"
  ),
  wkt = "geom", crs = 4326
)

opt_param_mod11a1 <- mf_get_opt_param("MOD11A1.061", roi)
str(opt_param_mod11a1)

# Now we can provide opt_param_mod11a1 as input parameter of the function mf_get_url().

time_ranges <- list(
  as.Date(c("2016-01-01", "2016-01-31")),
  as.Date(c("2017-01-01", "2017-01-31")),
  as.Date(c("2018-01-01", "2018-01-31")),
  as.Date(c("2019-01-01", "2019-01-31"))
)

(urls_mod11a1 <- map(.x = time_ranges, ~ mf_get_url(
  collection = "MOD11A1.061",
  variables = c("LST_Day_1km", "LST_Night_1km", "QC_Day", "QC_Night"),
  roi = roi,
  time_range = .x,
  opt_param = opt_param_mod11a1
)))

## End(Not run)
```

---

mf_get_url *Build the URL(s) of the data to download*

Description

Builds the OPeNDAP URL(s) of the spatiotemporal datacube to download, given a collection, variables, region and time range of interest.

Usage

```
mf_get_url(
    collection,
    variables = NULL,
    roi,
    time_range,
    output_format = "nc4",
    single_netcdf = TRUE,
    opt_param = NULL,
    credentials = NULL,
    verbose = "inform"
  )
```

Arguments

`collection` string. mandatory. Collection of interest (see details of mf_get_url).

`variables` string vector. optional. Variables to retrieve for the collection of interest. If not specified (default) all available variables will be extracted (see details of mf_get_url).

`roi` object of class `sf`. mandatory. Area of region of interest. Must be a Simple feature collection with geometry type POLYGON, composed of one or several rows (i.e. one or several ROIs), and with at least two columns: 'id' (an identifier for the roi) and 'geom' (the geometry).

`time_range` date(s) / POSIXlt of interest . mandatory. Single date/datetime or time frame : vector with start and end dates/times (see details).

`output_format` string. Output data format. optional. Available options are : "nc4" (default), "ascii", "json"

`single_netcdf` boolean. optional. Get the URL either as a single file that encompasses the whole time frame (TRUE) or as multiple files (1 for each date) (FALSE). Default to TRUE. Currently enabled only for MODIS and VIIRS collections.

`opt_param` list of optional arguments. optional. (see details).

`credentials` vector string of length 2 with username and password. optional if the function mf_login was previously executed.

`verbose` string. Verbose mode ("quiet", "inform", or "debug"). Default "inform".

Details

Argument `collection` : Collections available can be retrieved with the function mf_list_collections

Argument `variables` : For each collection, variables available can be retrieved with the function mf_list_variables

Argument `time_range` : Can be provided either as i) a single date (e.g. `as.Date("2017-01-01")`) or ii) a time frame provided as two bounding dates (starting and ending time) (e.g. `as.Date(c("2010-01-01","2010-01-30` or iii) a POSIXlt single time (e.g. `as.POSIXlt("2010-01-01 18:00:00")`) or iv) a POSIXlt time range (e.g. `as.POSIXlt(c("2010-01-01 18:00:00","2010-01-02 09:00:00"))`) for the half-hourly collection (GPM_3IMERGHH.06). If POSIXlt, hours must be provided in GMT.

Argument `single_netcdf` : for MODIS and VIIRS products from LP DAAC: download the data as a single file encompassing the whole time frame (TRUE) or as multiple files : one for each date, which is the behavious for the other collections - GPM and SMAP) (FALSE) ?

Argument opt_param : list of parameters related to the queried OPeNDAP server and the roi. See mf_get_opt_param for additional details. This list can be retrieved outside the function with the function mf_get_opt_param. If not provided, it will be automatically calculated within the mf_get_url function. However, providing it fastens the processing time. It might be particularly useful to precompute it with mf_get_opt_param in case the function is used within a loop for a single ROI.

Argument credentials : Login to the OPeNDAP servers is required to use the function. Login can be done either within the function or outside with the function mf_login

Value

a data.frame with one row for each dataset to download and 5 columns :

id_roi Identifier of the ROI

time_start Start Date/time for the dataset

collection Name of the collection

name Indicative name for the dataset

url https OPeNDAP URL of the dataset

maxFileSizeEstimated Maximum estimated data size for the dataset (in bites)

Examples

```
## Not run:

### First login to EOSDIS Earthdata with username and password.
# To create an account go to : https://urs.earthdata.nasa.gov/.
username <- "earthdata_un"
password <- "earthdata_pw"
log <- mf_login(credentials = c(username, password))

### Get the URLs to download the following datasets :
# MODIS Terra LST Daily (MOD11A1.061) (collection)
# Day + Night bands (LST_Day_1km,LST_Night_1km) (variables)
# over a 50km x 70km region of interest (roi)
# for the time frame 2017-01-01 to 2017-01-30 (30 days) (time_range)

roi <- sf::st_as_sf(
  data.frame(
    id = "roi_test",
    geom = "POLYGON ((-5.82 9.54, -5.42 9.55, -5.41 8.84, -5.81 8.84, -5.82 9.54))"
  ),
  wkt = "geom", crs = 4326
)

time_range <- as.Date(c("2017-01-01", "2017-01-30"))

(urls_mod11a1 <- mf_get_url(
  collection = "MOD11A1.061",
  variables = c("LST_Day_1km", "LST_Night_1km"),
  roi = roi,
  time_range = time_range
))

## Download the data :

res_dl <- mf_download_data(urls_mod11a1)

## Import as terra::SpatRast

modis_ts <- mf_import_data(dirname(res_dl$destfile[1]), collection = "MOD11A1.061")

## Plot the data

terra::plot(modis_ts)

## End(Not run)
```

mf_import_data *Import datasets downloaded using* modisfast *as a* terra::SpatRaster *object*

Description

Import datasets downloaded using `modisfast` as a `terra::SpatRaster` object

Usage

```
mf_import_data(
  path,
  collection,
  output_class = "SpatRaster",
  proj_epsg = NULL,
  roi_mask = NULL,
  vrt = FALSE,
  verbose = "inform",
  ...
)
```

Arguments

`path` character string. mandatory. The path to the local directory where the data are stored.

`collection` string. mandatory. Collection of interest (see details of mf_get_url).

`output_class` character string. Output object class. Currently only "SpatRaster" implemented.

`proj_epsg` numeric. EPSG of the desired projection for the output raster (default : source projection of the data).

`roi_mask` `SpatRaster` or `SpatVector` or `sf`. Area beyond which data will be masked. Typically, the input ROI of mf_get_url (default : NULL (no mask)) `vrt` boolean. Import virtual raster instead of SpatRaster. Useful for very large files. (default : FALSE)

`verbose` string. Verbose mode ("quiet", "inform", or "debug"). Default "inform".

`...` not used

Value

a `terra::SpatRast` object

Note

Although the data downloaded through `modisfast` could be imported with any netcdf-compliant R package (`terra`, `stars`, `ncdf4`, etc.), care must be taken. In fact, depending on the collection, some "issues" were raised. These issues are independent from `modisfast` : they result most of time of a lack of full implementation of the OPeNDAP framework by the data providers. Namely, these issues are :

- for MODIS and VIIRS collections : CRS has to be provided
- for GPM collections : CRS has to be provided + data have to be flipped

The function mf_import_data includes the processing that needs to be done at the data import phase in order to safely use the data as `terra` objects.

Also note that reprojecting over large ROIs using the argument `proj_epsg` might take long. In this case, setting the argument `vrt` to TRUE might be a solution.

Examples

```
## Not run:

### Login to EOSDIS Earthdata with your username and password
log <- mf_login(credentials = c("earthdata_un", "earthdata_pw"))

### Set-up parameters of interest
coll <- "MOD11A1.061"

bands <- c("LST_Day_1km", "LST_Night_1km")

time_range <- as.Date(c("2017-01-01", "2017-01-30"))

roi <- sf::st_as_sf(
  data.frame(
    id = "roi_test",
    geom = "POLYGON ((-5.82 9.54, -5.42 9.55, -5.41 8.84, -5.81 8.84, -5.82 9.54))"
  ),
  wkt = "geom", crs = 4326
)

### Get the URLs of the data
(urls_mod11a1 <- mf_get_url(
  collection = coll,
  variables = bands,
  roi = roi,
  time_range = time_range
))

### Download the data
res_dl <- mf_download_data(urls_mod11a1)

### Import the data as terra::SpatRast
modis_ts <- mf_import_data(dirname(res_dl$destfile[1]), collection = coll)

### Plot the data
terra::plot(modis_ts)

## End(Not run)
```

mf_list_collections — *Get the collections available for download with the* `modisfast` *package*

Description

Get the collections available for download using the package and a set of related information

Usage

```
mf_list_collections()
```

Value

A data.frame with the collections available, and a set of related information for each one. Main columns are :

collection Collection short name

source Data provider

long_name Collection long name

doi DOI of the collection

start_date First available date for the collection

url_opendapserver URL of the OPeNDAP server of the data

Examples

```
(head(mf_list_collections()))
```

mf_list_variables *Get information for the variables (bands) available for a given collection*

Description

Get the variables available for a given collection, along with a set of related information for each.

Usage

```
mf_list_variables(collection, credentials = NULL, verbose = "inform")
```

Arguments

`collection`	string. mandatory. Collection of interest (see details of mf_get_url).
`credentials`	vector string of length 2 with username and password. optional if the function mf_login was previously executed.
`verbose`	string. Verbose mode ("quiet", "inform", or "debug"). Default "inform".

Value

A data.frame with the variables available for the collection, and a set of related information for each variable. The variables marked as "extractable" in the column "extractable_with_modisfast" can be provided as input parameter `variables` of the function mf_get_url

Examples

```
## Not run:
# login to Earthdata
log <- mf_login(c("earthdata_un", "earthdata_pw"))

# Get the variables available for the collection MOD11A1.061
(df_varinfo <- mf_list_variables("MOD11A1.061"))

## End(Not run)
```

mf_login *Login to EOSDIS EarthData account*

Description

Login to EOSDIS EarthData before querying servers and download data

Usage

```
mf_login(credentials, verbose = "inform")
```

Arguments

credentials	vector string of length 2 with username and password. optional if the function mf_login was previously executed.
verbose	string. Verbose mode ("quiet", "inform", or "debug"). Default "inform".

Details

An EOSDIS EarthData account is mandatory to download the data. You can create a free account here : `https://urs.earthdata.nasa.gov/`.

Value

None.

Examples

```
## Not run:
username <- "earthdata_un"
password <- "earthdata_pw"
mf_login(credentials = c(username, password))

## End(Not run)
```

mf_modisfast *Download (and possibly import) MODIS, VIIRS and GPM Earth Observation data*

Description

Download and possibly import MODIS, VIIRS and GPM Earth Observation data quickly and efficiently. This function is a wrapper for mf_login, mf_get_url, mf_download_data and mf_import_data. Whenever possible, users should prefer executing the functions mf_login, mf_get_url, mf_download_data and mf_import_data sequentially rather than using this high-level function

Usage

```
mf_modisfast(
  collection,
  variables,
  roi,
  time_range,
  path = tempfile("modisfast_"),
  earthdata_username,
  earthdata_password,
  parallel = FALSE,
  verbose = "inform",
  import = TRUE,
  ...
)
```

Arguments

`collection` string. mandatory. Collection of interest (see details of mf_get_url).

`variables` string vector. optional. Variables to retrieve for the collection of interest. If not specified (default) all available variables will be extracted (see details of mf_get_url).

`roi` object of class `sf`. mandatory. Area of region of interest. Must be a Simple feature collection with geometry type POLYGON, composed of one or several rows (i.e. one or several ROIs), and with at least two columns: 'id' (an identifier for the roi) and 'geom' (the geometry).

`time_range` date(s) / POSIXlt of interest . mandatory. Single date/datetime or time frame : vector with start and end dates/times (see details).

`path` string. Target folder for the data to download. Default : temporary folder.

`earthdata_username` EarthData username

`earthdata_password` EarthData username

`parallel` boolean. Parallelize the download ? Default to FALSE

`verbose` string. Verbose mode ("quiet", "inform", or "debug"). Default "inform".

`import` boolean. Import the data as a SpatRast object ? default TRUE. FALSE will download the data but not import them it in R.

`...` Further arguments to be passed to mf_import_data

Value

if the parameter `import` is set to TRUE, a `terra::SpatRast` object ; else a data.frame providing details of the data downloaded (see output of mf_download_data).

See Also

mf_login, mf_get_url, mf_download_data, mf_import_data

Examples

```
## Not run:

### Set-up parameters of interest
coll <- "MOD11A1.061"

bands <- c("LST_Day_1km", "LST_Night_1km")

time_range <- as.Date(c("2017-01-01", "2017-01-30"))
```

```
roi <- sf::st_as_sf(
  data.frame(
    id = "roi_test",
    geom = "POLYGON ((-5.82 9.54, -5.42 9.55, -5.41 8.84, -5.81 8.84, -5.82 9.54))"
  ),
  wkt = "geom", crs = 4326
)

### Download and import the data
modis_ts <- mf_modisfast(
  collection = coll,
  variables = bands,
  roi = roi,
  time_range = time_range,
  earthdata_username = "earthdata_un",
  earthdata_password = "earthdata_pw"
 )

### Plot the data
terra::plot(modis_ts)

## End(Not run)
```

Index