Package: ERA5Flux (via r-universe)

October 21, 2026

Title Improve the Utility of ERA5-Land Data at AmeriFlux Site Locations

Version 1.0.0

Description Collection of tools useful for workflows involving ERA5-Land and AmeriFlux data. Includes functions for downloading ERA5-Land data based on AmeriFlux sites, formatting the downloaded data to match AmeriFlux standards, merging both datasets together to ensure consistent timestamps, and blending them to create a single gap-filled time series. For more details, visit the Climate Data Store's catalogue entry for ERA5-Land at <https://cds.climate.copernicus.eu/datasets/reanalysis-era5-land> and the AmeriFlux homepage at <https://ameriflux.lbl.gov>.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Imports curl, dplyr, ecmwfr, lubridate, lutz, ncdf4, purrr, readr, readxl, stats, terra

Suggests knitr, rmarkdown, sf

URL https://github.com/Malone-Disturbance-Ecology-Lab/ERA5Flux, https://malone-disturbance-ecology-lab.github.io/ERA5Flux/

BugReports https://github.com/Malone-Disturbance-Ecology-Lab/ERA5Flux/issues

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

Config/pak/sysreqs libgdal-dev gdal-bin libgeos-dev libsecret-1-dev libnetcdf-dev libssl-dev libproj-dev libsqlite3-dev libx11-dev

Repository https://malone-disturbance-ecology-lab.r-universe.dev

Date/Publication 2026-10-08 19:49:04 UTC

RemoteUrl https://github.com/malone-disturbance-ecology-lab/era5flux

RemoteRef HEAD

RemoteSha 180ceadccfc6fc877e3f18922b2a78c958382cd5

Contents

blend_ERA5_Flux 2
check_DST 4
date_conversion 5
day_padding 6
download_ERA5 6
get_land_proportion 8
get_land_sea_mask 9
get_site_metadata 10
merge_ERA5_Flux 11
netcdf_df_formatter 13
netcdf_to_csv 14
utc_offset 15

Index **16**

`blend_ERA5_Flux` *Blend ERA5-Land and AmeriFlux Data*

Description

This function is used to blend data from AmeriFlux and data from ERA5-Land, ensuring they both have the same start and end timestamps. Please first make sure to run the merge function (`merge_ERA5_Flux()`), because output of merge function will be used as input of this blending function.

Usage

```
blend_ERA5_Flux(
  merged_data = NULL,
  varname_FLUX = NULL,
  varname_ERA5 = NULL,
  blending_rule = NULL
)
```

Arguments

`merged_data`	(data.frame) A data frame that has a datetime stamp column named "time" with the format: "%Y-%m-%d %H:%M:%S". The time step of the "time" column is the same with that of AmeriFlux file. It also includes the columns of AmeriFlux and ERA5-Land data that were merged together.
`varname_FLUX`	(character) A vector of variable names in AmeriFlux BASE data to be blended with ERA5-Land data.
`varname_ERA5`	(character) A vector of variable names in ERA5-Land data to be blended with AmeriFlux BASE data.
`blending_rule`	(character) A vector of blending rules to use. There are four types of blending rules that can be used to blend AmeriFlux and ERA5-Land variables:

- "lm": Linear regression with slope. Fits a linear model with slope, FLUX ~ ERA5, then only fills missing values in FLUX with predicted values from ERA5.
- "lm_no_intercept": Linear regression without slope. Fits a linear model without slope, FLUX ~ ERA5, then only fills missing values in FLUX with predicted values from ERA5.
- "replace": Replace AmeriFlux variable with ERA5-Land variable.
- "automatic": Checks for non-missing FLUX values. If >=50% present then uses "lm" approach. If <50% present then fallback to "replace".

Value

(data.frame) A data frame with the following characteristics:

- Time step of the "time" column is the same with that of AmeriFlux file.
- It also includes the columns of `varname_FLUX`, the columns of `varname_ERA5`.
- Variable column with the blending rule applied on it. Name of the blended column is similar to AmeriFlux column name but with addition of "_f".

Note

Please note that the length of `varname_FLUX` must be the same as the length of `varname_ERA5`; at the same location, `varname_FLUX` and `varname_ERA5` should refer to the same variable despite the fact that AmeriFlux and ERA5-Land may use different names for the same variable. For example, for incoming shortwave radiation, ERA5-Land uses "ssrd", but AmeriFlux uses "SW_IN". Additionally, if you have multiple variables like precipitation and air temperature, you must specify a blending rule for each one.

Author(s)

Ammara Talib and Junna Wang

Examples

```
# Point to AmeriFlux CSV data
filename_FLUX <- system.file("extdata",
                             "example_AmeriFlux",
                             "AMF_US-GL2_BASE-BADM_2-5",
                             "AMF_US-GL2_BASE_HH_2-5.csv",
                             package = "ERA5Flux")

# Point to ERA5-Land CSV data
filename_ERA5 <- system.file("extdata",
                             "example_processed_ERA5",
                             "US_GL2_2024_2025_ssrd.csv",
                             package = "ERA5Flux")

# List AmeriFlux variable(s) to be merged with ERA5-Land
varname_FLUX <- c("SW_IN")
# List ERA5-Land variable(s) to be merged with AmeriFlux
```

```
varname_ERA5 <- c("ssrd")

# Run the merge function first, because its output will be used as input for this blending function
# Merge AmeriFlux and ERA5-Land data together
merged_data <- merge_ERA5_Flux(filename_FLUX, filename_ERA5, varname_FLUX, varname_ERA5)
head(merged_data)

# Specify the blending rule(s)
# If you have multiple variables, specify a rule for each variable
blending_rule <- c("replace")
# Blend AmeriFlux and ERA5-Land data together
merg_blend <- blend_ERA5_Flux(merged_data, varname_FLUX, varname_ERA5, blending_rule)
head(merg_blend)
```

---

check_DST *Check Daylight Saving Time*

Description

This function will check if a given time point is in Daylight Saving Time (DST). If so, it will adjust the time to the standard (non-DST) time.

Usage

```
check_DST(lat = NULL, lon = NULL, timepoint = NULL)
```

Arguments

lat	(numeric) Latitude coordinate in decimal degrees.
lon	(numeric) Longitude coordinate in decimal degrees.
timepoint	(character or POSIXt object) Time point in either %Y-%m-%d HH:MM:SS or %Y/%m/%d HH:MM:SS format.

Value

(list) A list containing a boolean representing the DST status and the standard time if DST is in effect.

Examples

```
# This will get the current time
timepoint <- lubridate::now()
# Check whether the time is in DST at latitude 25.4, longitude -80.5 and get the standard time if so
is_DST <- check_DST(lat = 25.4, lon = -80.5, timepoint)
is_DST
```

date_conversion *Date Conversion*

Description

This function will produce a converted time, either UTC -> local, or local -> UTC.

Usage

```
date_conversion(lat = NULL, lon = NULL, time = NULL, flag = NULL)
```

Arguments

lat	(numeric) Latitude coordinate in decimal degrees.
lon	(numeric) Longitude coordinate in decimal degrees.
time	(character) Time in %Y-%m-%d %H:%M:%S format.
flag	(numeric) Set flag to 0 for local to UTC, set flag to 1 for UTC to local.

Value

A POSIXct object.

Note

IT TAKES INTO ACCOUNT Daylight Saving Time!!

Author(s)

Boya ("Paul") Zhang

Examples

```
# Convert local to UTC
utc_result_winter <- date_conversion(25.2, -80.4, "2018-01-16 22:02:37", 0)
utc_result_winter

# Convert UTC to local
local_result <- date_conversion(25.2, -80.4, "2018-01-16 22:02:37", 1)
local_result
```

day_padding *Pad Dates by 1 Day*

Description

This function is designed to pad 1 day for the beginning and ending days. It will add 1 day to start and end dates for future time zone offset issues.

Usage

```
day_padding(start_day = NULL, end_day = NULL)
```

Arguments

start_day	(character) Start date in either %Y-%m-%d or %Y/%m/%d format.
end_day	(character) End date in either %Y-%m-%d or %Y/%m/%d format.

Value

(list) A list containing the new start and end dates in either %Y-%m-%d or %Y/%m/%d format, padded by 1 day each.

Author(s)

Boya ("Paul") Zhang

Examples

```
# Pad starting and ending dates by 1 day
result <- day_padding("2024-01-01", "2025-12-31")
result
```

download_ERA5 *Download ERA5-Land Data*

Description

This function uses the `ecmwfr` package to download ERA5-Land data for each site specified in the AmeriFlux site metadata data frame. The temporal and geographical coverage of the downloaded data will match the AmeriFlux site metadata data frame.

Usage

```
download_ERA5(
  my_key = NULL,
  site_metadata = NULL,
  mask = NULL,
  download_path = NULL
)
```

Arguments

`my_key`	(character) A secret ECMWF API key.
`site_metadata`	(data.frame) A data frame that has the AmeriFlux site metadata. Create the metadata with `get_site_metadata()`.
`mask`	(character) File path to the ERA5-Land land-sea mask. Get the mask with `get_land_sea_mask()`.
`download_path`	(character) Path to the folder where the ERA5-Land data will get downloaded to.

Value

(character) Paths to the downloaded files.

Note

If you haven't done so already, you may need to accept the data license agreement first before you can download the data. Visit the Copernicus Climate Data Store User Profile page at https://cds.climate.copernicus.eu/profile to accept the appropriate license(s). `download_ERA5()` also requires a valid Climate Data Store API key. When you're logged into the Copernicus Climate Data Store, you can grab your API key by clicking on your name for your account in the top right corner, scrolling down to the "API key" section, and copying the API key.

Author(s)

David Reed

Examples

```
# Specify the ERA5-Land variables you want to get
# Choose any combination from:
# 2m_temperature, total_precipitation, and surface_solar_radiation_downwards
my_variables <- c("surface_solar_radiation_downwards")

# Point to the folder containing the unzipped site folders and requested files manifest
# For the purposes of this example, an example data folder is used
# Please point to your own existing folder for your own workflow
my_AmeriFlux_folder <- system.file("extdata", "example_AmeriFlux", package = "ERA5Flux")

# Create the AmeriFlux site metadata
my_site_metadata <- get_site_metadata(folder = my_AmeriFlux_folder,
```

```
                                    selected_variables = my_variables)

# Paste your own ECMWF API key
# You can generate your own key by creating an account at the Copernicus Climate Data Store
# Please see the vignette for more info
my_key <- "my_own_ECMWF_key"


# Download the land-sea mask with get_land_sea_mask() if you haven't done so already
# Create a temporary directory to download to
temp_path <- tempdir()
# Download the land-sea mask
get_land_sea_mask(file_name = "lsm_1279l4_0.1x0.1.grb_v4_unpack.nc",
                  download_path = temp_path)

# Set the path to your land-sea mask
my_path_to_mask <- file.path(temp_path, "lsm_1279l4_0.1x0.1.grb_v4_unpack.nc")


# Point to the folder where you want the ERA5-Land data to download to
# For the purposes of this example, a temporary directory is used
# Please point to your own existing folder for your own workflow
my_ERA5_download_path <- tempdir()

## Not run:
# Download the corresponding ERA5-Land data
download_ERA5(my_key = my_key,
              site_metadata = my_site_metadata,
              mask = my_path_to_mask,
              download_path = my_ERA5_download_path)

## End(Not run)
```

---

get_land_proportion *Get Land Proportion*

Description

This function is designed to return land proportion given a latitude and longitude.

Usage

```
get_land_proportion(nc_file = NULL, lat = NULL, lon = NULL)
```

Arguments

nc_file	(character) File path to the .nc file.
lat	(numeric) Latitude coordinate in decimal degrees.
lon	(numeric) Longitude coordinate in decimal degrees.

Value

(data.frame) A data frame containing the land proportion.

Note

In cycles of the ECMWF Integrated Forecasting System (IFS) from CY41R1 (introduced in May 2015) onwards, grid boxes where this parameter has a value above 0.5 can be comprised of a mixture of land and inland water but not ocean. Grid boxes with a value of 0.5 and below can only be comprised of a water surface. In the latter case, the lake cover is used to determine how much of the water surface is ocean or inland water.

In cycles of the IFS before CY41R1, grid boxes where this parameter has a value above 0.5 can only be comprised of land and those grid boxes with a value of 0.5 and below can only be comprised of ocean. In these older model cycles, there is no differentiation between ocean and inland water.

Author(s)

Boya ("Paul") Zhang

Examples

```
# Create a temporary directory to download to
temp_path <- tempdir()
# Download the land-sea mask
get_land_sea_mask(file_name = "lsm_1279l4_0.1x0.1.grb_v4_unpack.nc",
                  download_path = temp_path)

# Get land proportion for a given latitude and longitude
get_land_proportion(nc_file = file.path(temp_path, "lsm_1279l4_0.1x0.1.grb_v4_unpack.nc"),
                    lat = 25.2,
                    lon = -80.3)
```

---

get_land_sea_mask *Get Land-Sea Mask from ERA5-Land*

Description

This function downloads the netCDF land-sea mask from ERA5-Land. The mask is needed in order to run `download_ERA5()`. The mask can be found online here: https://confluence.ecmwf.int/pages/viewpage.action?pageId=1 parameterlistingParameterlistings.

Usage

```
get_land_sea_mask(
  file_name = "lsm_1279l4_0.1x0.1.grb_v4_unpack.nc",
  download_path = getwd()
)
```

Arguments

`file_name` (character) Name of the land-sea mask. The default name is "lsm_1279l4_0.1x0.1.grb_v4_unpack.nc".

`download_path` (character) Path to an existing folder where the land-sea mask will get downloaded to. The default location is the current working directory.

Value

Path to the downloaded land-sea mask (invisibly).

Examples

```
# Create a temporary directory to download to
temp_path <- tempdir()
# Download the land-sea mask
get_land_sea_mask(file_name = "lsm_1279l4_0.1x0.1.grb_v4_unpack.nc",
                  download_path = temp_path)
```

`get_site_metadata` *Get AmeriFlux Site Metadata*

Description

This function is designed to build an AmeriFlux site metadata data frame. The data frame consists of site code, latitude, longitude, start date, end date, and ERA5-Land variables needed. This data frame will be used to create the ERA5-Land download request. To get started with this metadata function, you must have AmeriFlux data downloaded already. When downloading AmeriFlux data, select sites, download data and the "requested_files_manifest" text file. Then extract all files to a data folder, where the extracted data for each site has its own respective subfolder. ERA5-Land variables need to be added by the user. Please see the vignette for more help.

Usage

```
get_site_metadata(folder = NULL, selected_variables = NULL)
```

Arguments

`folder` (character) Path to the folder containing extracted AmeriFlux data and "requested_files_manifest" text file.

`selected_variables` (character) A vector of variable names. The possible options are: `2m_temperature`, `total_precipitation`, and `surface_solar_radiation_downwards`.

Value

(data.frame) A data frame containing the site metadata.

Author(s)

David Reed

Examples

```
# Specify the ERA5-Land variables you want to get
# Choose any combination from:
# 2m_temperature, total_precipitation, and surface_solar_radiation_downwards
my_variables <- c("surface_solar_radiation_downwards")

# Point to the folder containing the unzipped site folders and requested files manifest
# For the purposes of this example, an example data folder is used
# Please point to your own existing folder for your own workflow
my_AmeriFlux_folder <- system.file("extdata", "example_AmeriFlux", package = "ERA5Flux")

# Create the AmeriFlux site metadata
my_site_metadata <- get_site_metadata(folder = my_AmeriFlux_folder,
                                      selected_variables = my_variables)

my_site_metadata
```

---

merge_ERA5_Flux *Merge ERA5-Land and AmeriFlux Data*

Description

This function is used to merge data from AmeriFlux and data from ERA5-Land, ensuring they both have the same start and end timestamps.

Usage

```
merge_ERA5_Flux(
  filename_FLUX = NULL,
  filename_ERA5 = NULL,
  varname_FLUX = NULL,
  varname_ERA5 = NULL
)
```

Arguments

`filename_FLUX`	(character) The file path to a CSV file of AmeriFlux BASE data downloaded from https://ameriflux.lbl.gov/.
`filename_ERA5`	(character) The file path to a CSV file of ERA5-Land meterological data downloaded from https://cds.climate.copernicus.eu/datasets/reanalysis-era5-land. Please note that the original ERA5-Land files are in .nc format. You may want to convert these files into CSV format using the function `netcdf_to_csv()`.

varname_FLUX (character) A vector of variable names in AmeriFlux BASE data to be merged with ERA5-Land data.

varname_ERA5 (character) A vector of variable names in ERA5-Land data to be merged with AmeriFlux BASE data.

Value

(data.frame) A data frame with the following characteristics:

- Datetime stamp column named "time" with the format: "%Y-%m-%d %H:%M:%S".
- Time step of the "time" column is the same with that of AmeriFlux file.
- It also includes the columns of varname_FLUX, the columns of varname_ERA5.

Note

Please note that the length of varname_FLUX must be the same as the length of varname_ERA5; at the same location, varname_FLUX and varname_ERA5 should refer to the same variable despite the fact that AmeriFlux and ERA5-Land may use different names for the same variable. For example, for incoming shortwave radiation, ERA5-Land uses "ssrd", but AmeriFlux uses "SW_IN".

Author(s)

Ammara Talib and Junna Wang

Examples

```
# Point to AmeriFlux CSV data
filename_FLUX <- system.file("extdata",
                             "example_AmeriFlux",
                             "AMF_US-GL2_BASE-BADM_2-5",
                             "AMF_US-GL2_BASE_HH_2-5.csv",
                             package = "ERA5Flux")

# Point to ERA5-Land CSV data
filename_ERA5 <- system.file("extdata",
                             "example_processed_ERA5",
                             "US_GL2_2024_2025_ssrd.csv",
                             package = "ERA5Flux")

# List AmeriFlux variable(s) to be merged with ERA5-Land
varname_FLUX <- c("SW_IN")
# List ERA5-Land variable(s) to be merged with AmeriFlux
varname_ERA5 <- c("ssrd")

# Merge AmeriFlux and ERA5-Land data together
merged_data <- merge_ERA5_Flux(filename_FLUX, filename_ERA5, varname_FLUX, varname_ERA5)
head(merged_data)
```

netcdf_df_formatter *NetCDF Reformatter*

Description

Reformats ERA5-Land .nc data into a data frame.

Usage

```
netcdf_df_formatter(nc_file_path = NULL, site_lat = NULL, site_lon = NULL)
```

Arguments

nc_file_path	(character) File path to ERA5-Land netCDF file.
site_lat	(numeric) Latitude coordinate of site in decimal degrees.
site_lon	(numeric) Longitude coordinate of site in decimal degrees.

Value

(data.frame) Data frame of the following characteristics:

- Datetime stamp column named "time".
- UTC timezones converted to local time.
- Time column formatted as yyyyMMddHHmm, time zone determined using coordinates.
- Variables names from ERA5-Land dataset maintained.
- ERA5-Land units converted to AmeriFlux units:
  - Solar radiation (ssrd) from Jm-2 to Wm-2.
  - Air Temperature (t2m) from Kelvin to Celsius.
  - Total precipitation (tp) from meters to millimeters.

Examples

```
# Point to a netCDF file
nc_file_path <- system.file("extdata", "example_path_to_ERA5_download_folder",
                            "ERA5-US-GL2-2025-1.nc", package = "ERA5Flux")

# Specify the site latitude and longitude coordinates
site_lat <- 46.7167
site_lon <- -87.4

# Reformat the netCDF
result <- netcdf_df_formatter(nc_file_path, site_lat, site_lon)
head(result)
```

netcdf_to_csv *Export NetCDF to CSV*

Description

Takes a directory of ERA5-Land .nc data as an argument and exports the data in CSV format. This function grabs each netCDF file and runs `netcdf_df_formatter()` on it. It builds a list of variables across all data frames in the folder and joins data by time, with an option to filter to return only full years of data.

Usage

```
netcdf_to_csv(
  site_folder = NULL,
  output_filepath = NULL,
  site_name = NULL,
  site_lat = NULL,
  site_lon = NULL,
  full_year = FALSE
)
```

Arguments

`site_folder`	(character) A folder for one site with netCDF data. The netCDF files can be of different variables and of different years so long as it is for one site.
`output_filepath`	(character) File path to where the output CSV should be written.
`site_name`	(character) Name of the site that will be concatenated onto CSV file name (e.g. US_GL2).
`site_lat`	(numeric) Latitude coordinate of site in decimal degrees.
`site_lon`	(numeric) Longitude coordinate of site in decimal degrees.
`full_year`	(bool) If TRUE, filter to include only complete years, such that the data will start with the first hour of year and end with the last hour of a year. Otherwise, return data as-is. The default is FALSE.

Value

.csv file of netCDF data within the site folder. The .csv file has the file name format: siteID_startYear_endYear_variableName. For example, US_Ho1_2001_2020_tp_t2m.csv. Each CSV file starts from the first hour of a year (e.g., 2001-01-01 00:00) and ends with the last hour of a year (e.g., 2020-12-31 23:00) if full_year == TRUE.

Examples

```
# Point to a folder containing ERA5-Land .nc files
site_folder <- system.file("extdata", "example_path_to_ERA5_download_folder", package = "ERA5Flux")
# Create a temporary directory to export our output to
output_filepath <- tempdir()

# Specify a site name
site_name <- "US_GL2"
# Specify the site latitude and longitude coordinates
site_lat <- 46.7167
site_lon <- -87.4

# Convert netCDF data to a CSV file
netcdf_to_csv(site_folder, output_filepath, site_name, site_lat, site_lon, full_year = FALSE)

# Read the CSV back in
data <- read.csv(list.files(output_filepath, pattern = "US_GL2", full.names = TRUE))
head(data)
```

---

utc_offset *Coordinated Universal Time (UTC) Offset*

Description

Offset is the difference between UTC and local time without considering Daylight Saving Time. This function will produce an hour offset number converting local to UTC.

Usage

```
utc_offset(lat = NULL, lon = NULL)
```

Arguments

lat	(numeric) Latitude coordinate in decimal degrees.
lon	(numeric) Longitude coordinate in decimal degrees.

Value

(numeric) The UTC offset.

Examples

```
# Get the UTC offset at latitude 25.4, longitude -80.5
utc_offset(lat = 25.4, lon = -80.5)
```

Index

blend_ERA5_Flux, 2

check_DST, 4

date_conversion, 5
day_padding, 6
download_ERA5, 6

get_land_proportion, 8
get_land_sea_mask, 9
get_site_metadata, 10

merge_ERA5_Flux, 11

netcdf_df_formatter, 13
netcdf_to_csv, 14

utc_offset, 15