Package ‘skytrackr’

October 8, 2025

Type Package

Title A Sky Illuminance Location Tracker

Version 1.0

Maintainer Koen Hufkens <koen.hufkens@gmail.com>

Description Calculate geolocations by light using template matching.
The routine uses a calibration free optimization of a sky illuminance model to
determine locations robustly using a template matching approach,
as described by Ekstrom (2004) <https://nipr.repo.nii.ac.jp/records/2496>,
and behaviourly informed constraints (step-selection).

URL https://github.com/bluegreen-labs/skytrackr

BugReports https://github.com/bluegreen-labs/skytrackr/issues

Depends R (>= 4.2)

License AGPL-3

Encoding UTF-8

LazyData true

ByteCompile true

RoxygenNote 7.3.2

Imports skylight, circular, BayesianTools, cli, utils, memoise, stats,
rlang, sf, terra, geosphere, tidyr, dplyr, ggplot2, plotly,
patchwork, mapview

Suggests knitr, rmarkdown, bookdown, covr, testthat, multidplyr

VignetteBuilder knitr

NeedsCompilation no

Author Koen Hufkens [aut, cre] (ORCID:
<https://orcid.org/0000-0002-5070-8109>),
BlueGreen Labs [cph, fnd]

Repository CRAN

Date/Publication 2025-10-08 19:50:08 UTC

Contents

`cc876` *Migrate Technology Ltd demo data*

Description

Demo data for a single day of Common swift light logger data as read from a Migrate Technology Ltd .lux file using stk_read_lux().

Usage

```
cc876
```

Format

DataFrame

logger logger ID

date date

date_time date and time

hour decimal hour

lux light levels in lux

Details

The format is consistent with what is required by the skytrackr() routine.

`land` *Land area polygon*

Description

Vector polygon of world land areas to constrain model optimization.

Usage

```
land
```

Format

sf

MULTIPOLYGON sf multipolygon

`likelihood` *Log likelihood cost function*

Description

Main cost function used during optimization, combining both the fit of the illuminance data with the step-selection function.

Usage

```
likelihood(par, data, model, loc, roi, step_selection, ...)
```

Arguments

`par`	A vector of parameter values, including one for the uncertainty on the target values.
`data`	A nested data structure with validation data included.
`model`	A model to run with data and par settings.
`loc`	The previous modeled step location.
`roi`	A region of interest with valid sampling locations.
`step_selection`	A step selection function on the distance of a proposed move.
`...`	extra arguments to pass to the function

Value

The single log-likelihood cost of a proposed parameter set.

`log_lux` *Simulate illuminance value*

Description

Calculates log(lux) values for a give location, date, time and sky conditions.

Usage

```
log_lux(par, data, ...)
```

Arguments

`par`	Three parameters specifying the illuminance model.
`data`	A data frame with the required drivers for the illuminance model.
`...`	optional other parameters to forward

Value

Sky illuminance as log(lux).

`read_deg_lux` *Read lux and deg files*

Description

This function is wrapped by the ‘stk_read_lux()‘ function.

Usage

```
read_deg_lux(file, verbose = TRUE)
```

Arguments

`file`	A lux or deg file.
`verbose`	provide detailed feedback

Value

A skytrackr data frame with logger data.

skytrackr	*Sky (illuminance) location estimation routine*

Description

Skytrack compares geolocator based light measurements in lux with those modelled by the sky illuminance model of Janiczek and DeYoung (1987).

Usage

```
skytrackr(
  data,
  start_location,
  tolerance = 1500,
  range = c(0.09, 148),
  scale = log(c(1e-05, 50)),
 control = list(sampler = "DEzs", settings = list(burnin = 250, iterations = 3000,
    message = FALSE)),
  mask,
  step_selection,
  plot = TRUE,
  verbose = TRUE
)
```

Arguments

`data`	A skytrackr data frame.
`start_location`	A start location of logging as a vector of latitude and longitude
`tolerance`	Tolerance distance on the search window for optimization, given in km (left/right, top/bottom). Sets a hard limit on the search window regardless of the step selection function used.
`range`	Range of values to consider during processing, should be provided in lux c(min, max) or the equivalent if non-calibrated.
`scale`	Scale / sky condition factor, by default covering the skylight() range of 1-10 (from clear sky to extensive cloud coverage) but can be extended for more flexibility to account for coverage by plumage, note that in case of non-physical accurate lux measurements values can have a range starting at 0.0001 (a multiplier instead of a divider). Values need to be provided on a log scale (default = log(c(0.00001, 50)))
`control`	Control settings for the Bayesian optimization, generally should not be altered (defaults to a Monte Carlo method). For detailed information I refer to the BayesianTools package documentation.
`mask`	Mask to constrain positions to land
`step_selection`	A step selection function on the distance of a proposed move, step selection is specified on distance (in km) basis.

`plot`	Plot a map during location estimation (updated every seven days)
`verbose`	Give feedback including a progress bar (TRUE or FALSE)

Details

Model fits are applied by default to values up to sunrise or after sunset only as most critical to the model fit (capturing daylength, i.e. latitude and the location of the diurnal pattern - longitudinal displacement).

Value

A data frame with location estimate, their uncertainties, and ancillary model parameters useful in quality control.

Examples

```
# define land mask with a bounding box
# and an off-shore buffer (in km), in addition
# you can specify the resolution of the resulting raster
mask <- stk_mask(
  bbox  =  c(-20, -40, 60, 60), #xmin, ymin, xmax, ymax
  buffer = 150, # in km
  resolution = 0.5 # map grid in degrees
  )

  # define a step selection distribution/function
  ssf <- function(x, shape = 0.9, scale = 100, tolerance = 1500){
  norm <- sum(stats::dgamma(1:tolerance, shape = shape, scale = scale))
  prob <- stats::dgamma(x, shape = shape, scale = scale) / norm
  }

# estimate locations
locations <- cc876 |> skytrackr(
  plot = TRUE,
  mask = mask,
  step_selection = ssf,
  start_location = c(50, 4),
      control = list(
        sampler = 'DEzs',
        settings = list(
        iterations = 10, # change iterations
         message = FALSE
        )
      )
  )
```

`stk_cluster` *Cluster geolocator co-variates*

Description

Uses k-means and hierarchical clustering to group geolocator covariates into consistent groups for visual analysis

Usage

```
stk_cluster(df, k = 2, method = "kmeans")
```

Arguments

`df`	A skytrackr data frame.
`k`	The number of k-means/hierarchical clusters to consider.
`method`	The method to use, "kmeans" (default), "hclust" can be set.

Value

The original data frame with attached cluster labels.

`stk_fit` *Fit illuminance (lux) profile*

Description

Fits a simulated lux profile to observed light logger data to estimate locations (parameters).

Usage

```
stk_fit(data, roi, loc, scale, control, step_selection)
```

Arguments

`data`	A skytrackr data frame
`roi`	A region of interest defined by a dynamic bounding box (set via the tolerance value and relative to the previous step)
`loc`	The location of the previous step
`scale`	Scale / sky condition factor covering the skylight() range of 1-10 (from clear sky to extensive cloud coverage) but can be extended for more flexibility to account for coverage by plumage, note that in case of non-physical accurate lux measurements values can have a range starting at 0.0001 (a multiplier instead of a divider).

`control`	Control settings for the Bayesian optimization, generally should not be altered (defaults to a Monte Carlo method). For detailed information I refer to the BayesianTools package documentation.
`step_selection`	A step selection function on the distance of a proposed move, step selection is specified on distance (in km) basis.

Value

An estimated illuminance based location (and its uncertainties).

`stk_map` *Plot skytrackr results*

Description

Create a map of estimated locations as a static or dynamic map.

Usage

```
stk_map(df, bbox, start_location, roi, dynamic = FALSE)
```

Arguments

`df`	A data frame with locations produced with the skytrackr() function
`bbox`	A geographic bounding box provided as a vector with the format xmin, ymin, xmax, ymax.
`start_location`	A start location as lat/lon to indicate the starting position of the track (optional)
`roi`	A region of interest under consideration, only used in plots during optimization
`dynamic`	Option to create a dynamic interactive graph rather than a static plot. Both the path as the locations are shown. The size of the points is proportional to the latitudinal uncertainty, while equinox windows are marked with red points. (default = FALSE)

Value

A ggplot map of tracked locations or mapview dynamic overview.

Examples

```
# define land mask with a bounding box
# and an off-shore buffer (in km), in addition
# you can specify the resolution of the resulting raster
mask <- stk_mask(
  bbox  =  c(-20, -40, 60, 60), #xmin, ymin, xmax, ymax
  buffer = 150, # in km
  resolution = 0.5 # map grid in degrees
  )

  # define a step selection distribution/function
  ssf <- function(x, shape = 0.9, scale = 100, tolerance = 1500){
  norm <- sum(stats::dgamma(1:tolerance, shape = shape, scale = scale))
  prob <- stats::dgamma(x, shape = shape, scale = scale) / norm
}

# estimate locations
locations <- cc876 |> skytrackr(
  plot = TRUE,
  mask = mask,
  step_selection = ssf,
  start_location = c(50, 4),
      control = list(
        sampler = 'DEzs',
        settings = list(
        iterations = 10, # change iterations
         message = FALSE
        )
      )
  )

#----- actual plotting routines ----
# static plot, with required bounding box
locations |> stk_map(bbox = c(-20, -40, 60, 60))

# dynamic plot
locations |> stk_map(dynamic = TRUE)
```

`stk_mask` *Generate a land surface mask*

Description

Returns a (buffered) land mask to constrain potential model results.

Usage

```
stk_mask(buffer = 0, resolution = 1, bbox, sf = FALSE)
```

Arguments

`buffer`	The buffer distance from land areas (in km, default = 0 excluding all water bodies).
`resolution`	The resolution of the spatial grid in degrees, when exporting as a terra SpatRaster (default = 1).
`bbox`	A bounding box of the mask to constrain the estimated location parameter space.

`sf`	Return the land mask as an 'sf' polygon, not a rasterized map for. use in map plotting, not used for processing (default = FALSE)

Value

A buffered land mask as an 'sf' or 'terra' map object.

Examples

```
# define land mask with a bounding box
# and an off-shore buffer (in km), in addition
# you can specifiy the resolution of the resulting raster
mask <- stk_mask(
  bbox  =  c(-20, -40, 60, 60), #xmin, ymin, xmax, ymax
  buffer = 150, # in km
  resolution = 0.5 # map grid in degrees
  )
```

`stk_profile` *Plot seasonal profiles*

Description

Provides static or dynamic (plotly) seasonal profile plot

Usage

```
stk_profile(data, logger, range = c(0, 1e+05), center = "day", plotly = FALSE)
```

Arguments

`data`	A skytrackr compatible data frame.
`logger`	The logger to plot.
`range`	The light range to plot.
`center`	Set the data to center data on "day" or "night" (default = "day").
`plotly`	Logical, convert to dynamic plotly plot or not (default = FALSE)

Value

A static or dynamic graph of light levels for a given logger.

`stk_read_glf` *Read Swiss Ornithology institute GLF files*

Description

Read Swiss Ornithology institute files in the ‘.glf‘ and re-formats them to a skytrackr compatible format.

Usage

```
stk_read_glf(files, verbose = TRUE)
```

Arguments

`files`	A ‘.glf‘ file or list of ‘.glf‘ files with light level values.
`verbose`	provide detailed feedback

Value

A skytrackr compatible data frame for use in further location estimation.

Examples

```
## Not run:
df <- stk_read_glf("your_SOI_glf_file.glf")

## End(Not run)
```

`stk_read_lux` *Read Migrate Technology .lux files*

Description

Read Migrate Technology Ltd. ‘.lux‘ files and re-formats them to a skytrackr compatible format.

Usage

```
stk_read_lux(files, verbose = TRUE)
```

Arguments

`files`	A ‘.lux‘ file or list of ‘.lux‘ files with light level values
`verbose`	provide detailed feedback

Value

A skytrackr compatible data frame for use in further location estimation.

Examples

```
# read in the demo lux file
df <- stk_read_lux(
 system.file("extdata/cc876.lux", package="skytrackr")
)
```

`stk_screen_twl` *Twilight screening routine*

Description

Removes poor quality data based on twilight heuristics. Allows for quick screening of data containing "false" twilight values.

Usage

```
stk_screen_twl(df, threshold = 1.5, dips = 3, step = 100, filter = TRUE)
```

Arguments

`df`	A skytrackr data frame.
`threshold`	A twilight threshold (default = 1.5).
`dips`	The allowed number of interruptions during a daylight profile below the twilight threshold before flagging as a poor quality "suspect" day.
`step`	A threshold of the allowed step change in illuminance values between the twilight value and the preceding one. Large jumps and the lack of a smooth transition suggest a false twilight (bird leaving a dark nest site long after or long before dawn or dusk).
`filter`	Logical if to return data pre-filtered, removing all poor quality days or false twilight ones (default = TRUE)

Value

A skytrackr data frame with poor twilight quality days removed and dusk and dawn timings marked (data is returned as a long format, not a wide format).

Examples

```
# set demo values artificially low as a demonstration
library(dplyr)
df <- cc876 |>
  mutate(
  value = ifelse(
    date_time > "2021-08-15 05:00:00" & date_time < "2021-08-15 12:00:00",
    0.1,
    value)
  )
```

```
# screen values and remove them (filter = TRUE)
df <- df |> stk_screen_twl(filter = TRUE)
```

Index