Package: photobiologyFlecks (via r-universe)

May 22, 2026

Type Package

Title Analysis of sun- and shade flecks

Version 0.0.0.9000

Date 2026-04-22

Description Functions for the analysis of time series of natural light irradiance measured at high frequency. Denoising, and quantification of duration and amplitude of sun-, shade-, cloud- and wind flecks.

License GPL (>= 2)

ByteCompile true

LazyLoad true

LazyData true

Encoding UTF-8

URL https://docs.r4photobiology.info/photobiologyFlecks/

BugReports https://github.com/aphalo/photobiologyFlecks/issues/

RoxygenNote 7.3.3

Requires R (>= 4.4.0)

Imports utils, stats, dplyr (>= 1.1.0), anytime (>= 0.3.10), lubridate (>= 1.9.4), SunCalcMeeus (>= 0.1.3)

Suggests ggplot2 (>= 3.5.0), photobiology (>= 0.14.0), photobiologyInOut (>= 0.4.30), knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

Config/Needs/website rmarkdown

Repository https://aphalo.r-universe.dev

Date/Publication 2026-04-22 15:09:50 UTC

RemoteUrl https://github.com/aphalo/photobiologyFlecks

RemoteRef HEAD

RemoteSha 27dcddfe67334653ce371b31af135208166377bb

Contents

`photobiologyFlecks-package`

photobiologyFlecks: Analysis of sun- and shade flecks

Description

Functions for the analysis of time series of natural light irradiance measured at high frequency. Denoising, and quantification of duration and amplitude of sun-, shade-, cloud- and wind flecks.

Author(s)

Maintainer: Pedro J. Aphalo <pedro.aphalo@helsinki.fi> (ORCID)

Authors:

- Maxime Durand <maxime.durand@universite-paris-saclay.fr> (ORCID)

References

Durand M, Matule B, Burgess AJ, Robson TM. 2021. Sunfleck properties from time series of fluctuating light. *Agricultural and Forest Meteorology* **308-309**, 108554. doi:10.1016/j.agrformet.2021.108554

See Also

Useful links:

- https://docs.r4photobiology.info/photobiologyFlecks/
- Report bugs at https://github.com/aphalo/photobiologyFlecks/issues/

combine_integ *Combine time series differing in time step*

Description

Combine time-series from higher resolution to lower resolution.

Usage

```
combine_integ(time, var, new.integ.ms = 20)
```

Arguments

time	numeric Vector of times from the time-series (*x*-axis).
var	numeric Vector of observations from the time-series (y-axis).
new.integ.ms	numeric New integration time for the returned time-series. Use same units as main time-series.

Details

Timeseries of observations at different time steps, e.g. 10 ms and 30 ms are combined into a new data frame. Higher resolution needs to be a multiple of lower resolution.

Value

A data frame

denoise_chunks *Denoise differences within data frames*

Description

Replace running differences smaller than a threshold by zeros in selected columns of data frames.

Usage

```
denoise_chunks(
  data,
  time.name = "TIMESTAMP",
  qty.name = NULL,
  absolute.threshold = 0,
  relative.threshold = 0.05,
  range.baseline = 0,
  add.signs = FALSE,
  verbose = FALSE
)
```

Arguments

data	data.frame or list of data.frame objects Each data frame containing at least one column with time stamps, one column with a measured quantity, and one column of running differences for each measured quantity.
time.name	character vector of length one Name of the variable containing time stamps for the observations.
qty.name	character vector Name(s) of variable(s) in data containing values observed quantities. If qty.name = NULL, the default, all columns are retained.
absolute.threshold	numeric The largest difference values to ignore, i.e., to set to zero. Expressed as a change per second.
relative.threshold	numeric The multiplier to apply to the spread of x to obtain the largest difference values to ignore, i.e., to set to zero. Expressed as a change per second.
range.baseline	numeric An additional value included in the computation of the range of the observations. Set range.baseline = NA for the spread applied to relative.threshold to be computed only based on the observations, set range.baseline = 0 for the range to include zero, i.e., use a relative threshold relative to the maximum observation.
add.signs	logical Flag indicating if values returned by sign() on the de-noised differences are to be added to the returned data frame chunks.
verbose	logical Report data columns found. Useful for debugging.

Details

When searching for changes in the sign of differences we may need to discard small values introduced by "measurement noise". These functions replace differences smaller than a threshold by zeros. This approach is an alternative to smoothing, which can be difficult to implement for irregular time series.

The argument passed to data can be either a bare data.frame object or a list containing one or more data frames, such as that returned by split_chunks().

The argument passed to absolute.threshold is directly expressed as the smallest value of differences to be retained with any smaller differences replaced by zero. In contrast, the argument passed to relative.threshold is a multiplier applied to the spread of the observations, where the spread is the difference between the largest and the smallest observed value for a given variable in data plus range.baseline. The values of the two thresholds are combined, so that the largest of the two values is used. Setting either threshold equal to zero, forces the other the one to be always used. The threshold used is computed as

```
max(abs(diff(range(c(range.baseline, x), na.rm = TRUE))) * relative.threshold, absolute.threshold)
```

with differences in data smaller than the threshold, set to zero.

The intended use of absolute.threshold is to allow filtering out both zero or dark noise and gain noise in the observed data, i.e., to be able to apply a minimum denoising even in the complete absence of flecks, but otherwise apply a denoising relative to the value of the largest observation or relative to the spread of the observations.

Value

`denoise_chunks()` returns a copy of `data`, either a `data.frame` or a `list`. Each dataframe with each column of differences named in `qty.name`, if present, replaced by the value returned by function `denoise_diffs()` applied to it, and optionally with columns added with the result of calling `sign()` on the denoised differences.

See Also

`split_chunks()` and `check_colnames()`.

`find_flecks` *Find flecks*

Description

Detects and characterizes "flecks" in a time series of irradiances.

Usage

```
find_flecks(
  time,
  var,
  zero.lim = 5e-04,
  minTime = 0,
  minAmp = 0,
  minPdiff = 0,
  asymmetry = 1/4,
  trimCV = 0.05,
  asmMethod = c("mean", "max", "rm"),
  bounds = c(0, 1),
  timeSplit = 10,
  shadeflecks = FALSE,
  time.digits = 3,
  var.digits = 2,
  verbose = TRUE
)
```

Arguments

`time`	numeric Vector of times from the time-series (*x*-axis).
`var`	numeric Vector of observations from the time-series (y-axis).
`zero.lim`	numeric Limit for multiplication, only values higher than `zero.lim` are kept (removes noise and prevents recording zeroes when time-series is flat.).
`minTime`	numeric Flecks found with duration below `minTime` will be discarded. Keep at 0 to keep all flecks.

`minAmp`	numeric Flecks found with amplitude below `minAmp` will be discarded. Keep at 0 to keep all flecks.
`minPdiff`	numeric Flecks found with a percent difference between peak and baseline that is below `minPdiff` will be discarded. Keep at 0 to keep all flecks.
`asymmetry`	numeric Threshold value to qualify fleck as asymeric. Asymetry happens when the baseline have large difference in their value.
`trimCV`	Control trimming threshold. Fleck baselines are trimmed iteratively based on the coefficient of variation between two points at each baseline side.
`asmMethod`	character One of `"mean"`, `"max"`, `"rm"`. Decides what to do with asymetric flecks `"mean"` averages the two baselines, `"max"` keeps the largest baseline, `"rm"` discard the asymetric fleck.
`bounds`	numeric vector of length 2 For relative amplitude calculations, normalize between 0 and 1 by default.
`timeSplit`	integer Increase time-series data frequency by linear interpolation. A value of 10 usually guarantee accuracy.
`shadeflecks`	logical If true, run the function in *shadefleck* mode instead of in the default *sunfleck* mode, i.e., the function will find troughs instead of peaks in the data.
`time.digits, var.digits`	integer Argument passed to parameter `digits` in internal calls to `round()` for `time` and `var` values.
`verbose`	logical If `TRUE`, provides more information while running.

Details

A sunfleck is characterized by an increase followed by a decrease in irradiance. As a first step zero crossings of the derivative are located using the same code as in `find_zeros()`. In a second step, flecks are searched and once found, are checked for asymmetry between baselines. If found, tries to extend baseline a bit. If still asymmetric, behaviour is as defined by `asmMethod`. Then fleck is compared againts criteria given by `minTime`, `minAmp` and `minPdiff`. If passed, the baselines are trimmed. Conditions are checked once more, and trimming is reversed if conditions are not passed any more. In the last step overlapping flecks are removed and the time interval between successive flecks is computed. Finally, the fleck properties are returned in a data frame.

References

Durand M, Matule B, Burgess AJ, Robson TM. 2021. Sunfleck properties from time series of fluctuating light. *Agricultural and Forest Meteorology* **308-309**, 108554. doi:10.1016/j.agrformet.2021.108554

`find_zeros` *Find all time points when time series crosses zero*

Description

First function to use on discrete time-series. Returns a vector giving the position of every zero crossing within the time-series.

Usage

```
find_zeros(time, var, zero.lim = 5e-04, timeSplit = 10, return_n1n2 = FALSE)
```

Arguments

`time`	numeric Vector of times from the time-series (*x*-axis).
`var`	numeric Vector of observations from the time-series (y-axis).
`zero.lim`	numeric Limit for multiplication, only values higher than `zero.lim` are kept (removes noise and prevents recording zeroes when time-series is flat.).
`timeSplit`	numeric Increase time-series frequency (value at t = 0 are copied from t = 1 to t = 9, etc.). 10 usually garantees accuracy.
`return_n1n2`	logical If `TRUE`, also return the value of the multiplication.

Details

For every time point, the numerical derivative is calculated using `diff()` and each sequential time point of the numerical derivative is multiplied. When the result is negative, the time-series crosses zero.

Value

An `integer` vector or a data frame with two variables.

References

Durand M, Matule B, Burgess AJ, Robson TM. 2021. Sunfleck properties from time series of fluctuating light. *Agricultural and Forest Meteorology* **308-309**, 108554. doi:10.1016/j.agrformet.2021.108554

`split_chunks` *Split data frame into chunks*

Description

Split a time series stored in a data frame at breaks (long time steps), returning a list of data frames or data chunks.

Usage

```
split_chunks(
  data,
  time.name = "TIMESTAMP",
  qty.name = NULL,
  time.step = NULL,
  chunk.min.time,
  chunk.min.rows = 2,
  add.diffs = TRUE,
```

```
  verbose = FALSE,
  na.rm = TRUE
)
```

Arguments

data	data.frame Containing at least one coloumn with time stamps and one column with a measured quantity.
time.name	character vector of length one Name of the variable containing time stamps for the observations.
qty.name	character vector Name(s) of variable(s) in data containing values observed quantities. If qty.name = NULL, the default, all columns are retained.
time.step	numeric The duration in seconds of one time step within a chunk. If NULL, the actual time steps are used.
chunk.min.time	numeric or duration Length of minimum time step length between data chunks. If numeric, expressed in seconds.
chunk.min.rows	integer The minimum number of rows that a chunk must have not to be discarded.
add.diffs	logical Flag indicating if values returned by diff() are to be added to the returned data frame chunks.
verbose	logical Report chunk names and lengths at each iteration. Useful for debugging.
na.rm	logical Omit rows of data containing NA values after selecting variables.

Details

When time series of data are acquired in bursts or chunks separated by longer time intervals it can be useful to extract the chunks into separate data frames before further analysis. This implementation does not assume the same duration for all chunks or the gaps, it searches for time intervals longer than a threshold duration and splits the data at these points. If the data contains no gaps, the whole data is returned as a single chunk.

When a minimum length for the individuals chunks is set with an argument to chunk.min.rows, chunks with fewer rows are discarded silently, unless verbose = TRUE.

With add.diffs = TRUE the running differences between values in the current row and the one above are added to the returned data frames. The value in the first row is NA for running differences, except for the time, in which case it is the time difference to the precceeding value in data.

Method diff() must be available for the class of the variable named by the argument to time.name. The class of this column is in most cases numeric, date, or time. If add.diffs = TRUE this requirement also applies to the variable(s) named by the argument passed to qty.name.

The number of chunks in the returned list of data frames and their lengths are reported in a message().

Value

A list of data frames of varying length, depending on the number of chunks found, possibly of length zero. The members of the list are named based on the starting time of each chunk. The variables included in the member data frames are those named by time.name and qty.name and optionally, their running differences.

summarize_chunks *Chunk summaries*

Description

Compute statistical summaries for each numeric variable in a time series chunk or in a list of time series chunks.

Usage

```
summarize_chunks(
  l,
  FUN = summarize_chunk,
  parameter.names = FUN(NULL, return.names = TRUE),
  add.times = FALSE,
  tz = "UTC",
  time.shift = 0,
  add.solar.times = add.times && !is.null(geocode),
  geocode = NULL,
  verbose = FALSE
)

summarize_chunk(x, return.names = FALSE)
```

Arguments

l	a named list of data frames.
FUN	function The function used to compute the summary of a numeric vector.
parameter.names	The names used to identify the members of the vector returned by FUN.
add.times	logical If TRUE, list names are converted into POSIX.ct and Date values and added in columns named time and date.
tz	character The time zone used to decode times.
time.shift	numeric A time shift expressed in hours. Only needed if original times do not match those at the time zone passed as argument to tz,
add.solar.times	logical If TRUE, add local solar time in column solar.time in addition to time and date.
geocode	A one row data.frame with columns lat and lon with geographical coordinates as numeric values in degrees W and N.
verbose	logical Report chunk names while walking through l. Useful for debugging.
x	numeric vector.
return.names	logical Return the names of the parameters as a character vector instead of the computed numeric values.

Details

A fixed set of summaries is computed for each numeric variable in a chunk and indexed by additional columns `parameter` and `chunk`.

If `add.times == TRUE` and the names in `l` are strings describing instants in time, they are decoded using functions `anytime()` and `anydate()` and added as columns `time` and `date`. If the argument passed to `l` was the list returned by `split_chunks()` the times and dates match those of the first time point in each chunk.

An argument passed to `time.shift` can be used to correct a consistent error in times such as a badly set clock during acquisition or when data acquisition times have been in UTC plus a constant time shift year round.

Value

A tibble with 10 rows for each chunk in the input, with one summary per row and one column for each numeric column in the chunks with their original names plus columns `chunk` and `parameter`, and if requested columns `time`, `date` and `solar.time` added.

A numeric vector of length ten, or a character vector of the same length.

`three_chunks.tb` *PAR irradiance time series*

Description

Time series of photosynthetically active radiation (PAR) measured at 50 ms time intervals in 15 min-long bursts every 30 min.

Usage

```
three_chunks.tb
```

Format

A `"data.frame"` object with 54700 rows.

Details

The `data.frame` named `three_chunks.tb` contains time stamps and PAR photon irradiances (PPFD).

The variables in each member spectrum are as follows:

- `time`
- `Q_PAR`

Examples

```
head(three_chunks.tb)
```

Index