Package 'GeoLight'

September 6, 2015

Type Package

Title Analysis of Light Based Geolocator Data

Version 2.0.0

Author Simeon Lisovski, Simon Wotherspoon, Michael Sumner, Silke Bauer, Tamara Emmenegger

Maintainer Simeon Lisovski <simeon.lisovski@gmail.com>

Description Provides basic functions for global positioning based on light intensity measurements over time. Positioning process includes the determination of sun events, a discrimination of residency and movement periods, the calibration of period-specific data and, finally, the calculation of positions.

License GPL (>= 2)

Depends R (>= 2.10), maps

Imports changepoint, fields, MASS,

LazyLoad yes

LazyData yes

NeedsCompilation no

Suggests knitr

VignetteBuilder knitr

Repository CRAN

Date/Publication 2015-09-06 16:21:36

R topics documented:

`GeoLight-package` *The GeoLight Package*

Description

This is a summary of all features of `GeoLight`, a `R`-package for analyzing light based geolocator data

Details

`GeoLight` is a package to derive geographical positions from daily light intensity pattern. Positioning and calibration methods are based on the threshold-method (Ekstrom 2004, Lisovski *et al.* 2012). A changepoint model from the `R` package `changepoint` is implemented to distinguish between periods of residency and movement based on the sunrise and sunset times. Mapping functions are implemented using the `R` package `maps`.

Getting Started

We refrain from giving detailed background on the (several steps of) analysis of light-based geolocator data here but strongly recommend the key-publications below.

Updates

We advise all users to update their installation of `GeoLight` regularly. Type `news(package="GeoLight")` to read news documentation about changes to the recent and all previous version of the package

Important notes

Most functions in GeoLight require the same initial units and mostly the format and object type is mandatory:

tFirst	yyyy-mm-dd hh:mm "UTC" (see: as.POSIXct, time zones)
tSecond	as *tFirst* (e.g. 2008-12-01 17:30)
type	either 1 or 2 depending on wheter *tFirst* is sunrise (1) or sunset (2)
coord	SpatialPoints or a matrix, containing x and y coordinates (in that order)
degElevation	a vector or a single value of sun elevation angle(s) in degrees (e.g. -6)

FUNCTIONS AND DATASETS

In the following, we give a summary of the main functions and sample datasets in the GeoLight package. Alternatively a list of all functions and datasets in alphabetical order is available by typing library(help=GeoLight). For further information on any of these functions, type help(function name).

CONTENTS

I. Determination of sunset and sunrise

gleTrans	transformation of already defined twilight events*
glfTrans	transformation of light intensity measurements over time*
luxTrans	transformation of light intensity measurements over time**
lightFilter	filter to remove noise in light intensity measurements during the night
twilightCalc	definition of twilight events (*sunrise, sunset*) from light intensity measurements

* written for data recorded by geolocator devices from the **Swiss Ornithological Institute**
** written for data recorded by geolocator devices from **Migrate Technology Ltd**

II. Residency Analysis

changeLight	function to distinguish between residency and movement periods
schedule	function to produce a data frame summerizing the residency and movement pattern

III. Calibration

See Lisovski *et al.* 2012 for all implemented calibration methods.

`getElevation`	function to calculate the sun elevation angle for data with known position
`HillEkstromCalib`	*Hill-Ekstrom calibration* for one or more defined stationary periods

IV. Positioning

`coord`	main function to derive a `matrix` of spatial coordinates
`distanceFilter`	filter function to reduce unrealistic positions (not recommended, since the filtering ignore positioning e
`loessFilter`	filter function to define outliers in sunrise and sunset times (defined twilight events)

V. Data visualisation

`tripMap`	function to map the derived positions and combine the coordinates in time order
`siteMap`	function to show the results of the residency analysis on a map

IV. Examples

`calib1`	data for calibration: light intensities
`calib2`	data for calibration: Calculated twilight events (from `calib1` by `twilightCalc`)
`hoopoe1`	light intensity measurements over time recorded on a migratory bird
`hoopoe2`	sunrise and sunset times: From light intensity measurement (from `hoopoe1`)

R Packages for Further Spatial Ananlyses

```
spatstat
adehabitat
gstat
trip
tripEstimation
move
...
```

Acknowledgements

Steffen Hahn, Felix Liechti, Fraenzi Korner-Nievergelt, Andrea Koelzsch, Eldar Rakhimberdiev, Erich Baechler, Eli Bridge, Andrew Parnell, Richard Inger

Authors

Simeon Lisovski, Simon Wotherspoon, Michael Sumner, Silke Bauer, Tamara Emmenegger
Maintainer: Simeon Lisovski <simeon.lisovski(at)gmail.com>

References

Ekstrom, P.A. (2004) An advance in geolocation by light. *Memoirs of the National Institute of Polar Research*, Special Issue, **58**, 210-226.

Fudickar, A.M., Wikelski, M., Partecke, J. (2011) Tracking migratory songbirds: accuracy of light-level loggers (geolocators) in forest habitats. *Methods in Ecology and Evolution*, DOI: 10.1111/j.2041-210X.2011.00136.x.

Hill, C. & Braun, M.J. (2001) Geolocation by light level - the next step: Latitude. *Electronic Tagging and Tracking in Marine Fisheries* (eds J.R. Sibert & J. Nielsen), pp. 315-330. Kluwer Academic Publishers, The Netherlands.

Hill, R.D. (1994) Theory of geolocation by light levels. *Elephant Seals: Population Ecology, Behavior, and Physiology* (eds L. Boeuf, J. Burney & R.M. Laws), pp. 228-237. University of California Press, Berkeley.

Lisovski, S. and Hahn, S. (2012) GeoLight - processing and analysing light-based geolocator data in R. *Methods in Ecology and Evolution*, doi: 10.1111/j.2041-210X.2012.00248.x

Lisovski, S., Hewson, C.M, Klaassen, R.H.G., Korner-Nievergelt, F., Kristensen, M.W & Hahn, S. (2012) Geolocation by light: Accuracy and precision affected by environmental factors. *Methods in Ecology and Evolution*, doi: 10.1111/j.2041-210X.2012.00185.x

Wilson, R.P., Ducamp, J.J., Rees, G., Culik, B.M. & Niekamp, K. (1992) Estimation of location: global coverage using light intensity. *Wildlife telemetry: remote monitoring and tracking of animals* (eds I.M. Priede & S.M. Swift), pp. 131-134. Ellis Horward, Chichester.

`calib1` *Example data for calibration: Light intensities and twilight events*

Description

Light intensity measurements over time (calib1) recorded at the rooftop of the Swiss Ornithological Institute (Lon: 8.0, Lat: 47.01). Defined twilight events from calib1 (calib2). These data serve as an example for calculating the sun elevation angle of an additional data set, which is subsequently used to calibrate the focal dataset.

References

Lisovski, S., Hewson, C.M, Klaassen, R.H.G., Korner-Nievergelt, F., Kristensen, M.W & Hahn, S. (2012) Geolocation by light: Accuracy and precision affected by environmental factors. *Methods in Ecology and Evolution*, DOI: 10.1111/j.2041-210X.2012.00185.x.

Examples

```
data(calib2)
 calib2$tFirst <- as.POSIXct(calib2$tFirst, tz = "GMT")
 calib2$tSecond <- as.POSIXct(calib2$tSecond, tz = "GMT")
getElevation(calib2, known.coord = c(8,47.01))
```

---

changeLight *Residency analysis using a changepoint model*

Description

Function to discriminate between periods of residency and movement based on consecutive sunrise and sunset data. The calculation is based on a changepoint model (**R** Package changepoint: cpt.mean) to find multiple changepoints within the data.

Usage

```
changeLight(tFirst, tSecond, type, twl, quantile = 0.9, rise.prob = NA,
  set.prob = NA, days = 5, plot = TRUE, summary = TRUE)
```

Arguments

tFirst	vector of sunrise/sunset times (e.g. 2008-12-01 08:30).
tSecond	vector of of sunrise/sunset times (e.g. 2008-12-01 17:30).
type	vector of either 1 or 2, defining tFirst as sunrise or sunset respectively.
twl	data.frame containing twilights and at least tFirst, tSecond and type (alternatively give each parameter separately).
quantile	probability threshold for stationary site selection. Higher values (above the defined quantile of all probabilities) will be considered as changes in the behavior. Argmuent will only be considered if either rise.prob and/or set.prob remain unspecified.
rise.prob	the probability threshold for **sunrise**: greater or equal values indicates changes in the behaviour of the individual.
set.prob	the probability threshold for **sunset**: higher and equal values indicates changes in the behaviour of the individual.
days	a threshold for the length of stationary period. Periods smaller than "days" will not be considered as a residency period
plot	logical, if TRUE a plot will be produced
summary	logical, if TRUE a summary of the results will be printed

Details

The cpt.mean from the R Package changepoint is a function to find a multiple changes in mean for data where no assumption is made on their distribution. The value returned is the result of finding the optimal location of up to Q changepoints (in this case as many as possible) using the cumulative sums test statistic.

Value

A `list` with probabilities for *sunrise* and *sunset* the user settings of the probabilities and the resulting stationary periods given as a `vector`, with the residency sites as positiv numbers in ascending order (0 indicate movement/migration).

Note

The sunrise and/or sunset times shown in the graph (if `plot=TRUE`) represent hours of the day. However if one or both of the twilight events cross midnight during the recording period the values will be formed to avoid discontinuity.

Author(s)

Simeon Lisovski & Tamara Emmenegger

References

Taylor, Wayne A. (2000) Change-Point Analysis: A Powerful New Tool For Detecting Changes.

M. Csorgo, L. Horvath (1997) Limit Theorems in Change-Point Analysis. *Wiley*.

Chen, J. and Gupta, A. K. (2000) Parametric statistical change point analysis. *Birkhauser*.

See Also

`changepoint`, `cpt.mean`

Examples

```
data(hoopoe2)
  hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
  hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
residency <- changeLight(hoopoe2, quantile=0.9)
```

---

coord *Simple Threshold Geolocation Estimates*

Description

Estimate location from consecutive twilights

Usage

```
coord(tFirst, tSecond, type, twl, degElevation = -6, tol = 0,
  method = "NOAA", note = TRUE)
```

Arguments

tFirst	vector of sunrise/sunset times (e.g. 2008-12-01 08:30).
tSecond	vector of of sunrise/sunset times (e.g. 2008-12-01 17:30).
type	vector of either 1 or 2, defining tFirst as sunrise or sunset respectively.
twl	data.frame containing twilights and at least tFirst, tSecond and type (alternatively give each parameter separately).
degElevation	the sun elevation angle (in degrees) that defines twilight (e.g. -6 for "civil twilight"). Either a single value, a vector with the same length as tFirst or nrow(x).
tol	tolerance on the sine of the solar declination (only implemented in method 'NOAA').
method	Defines the method for the location estimates. 'NOAA' is based on code and the excel spreadsheet from the NOAA site (http://www.esrl.noaa.gov/gmd/grad/solcalc/), 'Montenbruck' is based on Montenbruck, O. & Pfleger, T. (2000) Astronomy on the Personal Computer. *Springer*, Berlin.
note	logical, if TRUE a notation of how many positions could be calculated in proportion to the number of failures will be printed at the end.

Details

This function estimates the location given the times at which the observer sees two successive twilights.

Longitude is estimated by computing apparent time of local noon from sunrise and sunset, and determining the longitude for which this is noon. Latitude is estimated from the required zenith and the sun's hour angle for both sunrise and sunset, and averaged.

When the solar declination is near zero (at the equinoxes) latitude estimates are extremely sensitive to errors. Where the sine of the solar declination is less than tol, the latitude estimates are returned as NA.

The format (date and time) of *tFirst* and *tSecond* has to be "yyyy-mm-dd hh:mm" corresponding to Universal Time Zone UTC (see: as.POSIXct, time zones)

Value

A matrix of coordinates in decimal degrees. First column are longitudes, expressed in degrees east of Greenwich. Second column contains the latitudes in degrees north the equator.

Author(s)

Simeon Lisovski, Simon Wotherspoon, Michael Sumner

Examples

```
data(hoopoe2)
  hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
  hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
crds <- coord(hoopoe2, degElevation=-6, tol = 0.2)
## tripMap(crds, xlim=c(-20,20), ylim=c(5,50), main="hoopoe2")
```

`distanceFilter` *Filter for unrealistic positions within a track based on distance*

Description

The filter identifies unrealistic positions. The maximal distance per hour/day can be set corresponding to the particular species.

Usage

```
distanceFilter(tFirst, tSecond, type, twl, degElevation = -6, distance,
  units = "hour")
```

Arguments

`tFirst`	vector of sunrise/sunset times (e.g. 2008-12-01 08:30).
`tSecond`	vector of of sunrise/sunset times (e.g. 2008-12-01 17:30).
`type`	vector of either 1 or 2, defining `tFirst` as sunrise or sunset respectively.
`twl`	data.frame containing twilights and at least `tFirst`, `tSecond` and `type` (alternatively give each parameter separately).
`degElevation`	sun elevation angle in degrees (e.g. -6 for "civil twilight")
`distance`	the maximal distance in km per `units`. Distances above will be considered as unrealistic.
`units`	the time unite corresponding to the distance. Default is "hour", alternative option is "day".

Details

Note that this type of filter significantly depends on the calibration (`degElevation`). Especially during equinox periods. In contrast, the (`loessFilter`) is independent from positions (uses twilight times) and therefore superior.

Value

Logical `vector`. TRUE means the particular position passed the filter.

Author(s)

Simeon Lisovski, Fraenzi Korner-Nievergelt

geolight.convert *Convert GeoLight Format*

Description

Convert GeoLight data

Usage

```
geolight.convert(tFirst, tSecond, type)
```

Arguments

tFirst	times of first twilight.
tSecond	times of second twilight.
type	type of twilight.

Details

This function converts from the tFirst, tSecond format used by GeoLight to the twilight, rise format used by Stella and Estelle.

Value

A data frame with columns

twilight	times of twilight as POSIXct objects.
rise	logical vector indicating which twilights are sunrise.

getElevation *Calculate the appropriate sun elevation angle for known location*

Description

Function to calculate the median sun elevation angle for light measurements at a known location and the choosen light threshold.

Usage

```
getElevation(tFirst, tSecond, type, twl, known.coord, plot = TRUE,
  lnorm.pars = FALSE)
```

Arguments

tFirst	vector of sunrise/sunset times (e.g. 2008-12-01 08:30).
tSecond	vector of of sunrise/sunset times (e.g. 2008-12-01 17:30).
type	vector of either 1 or 2, defining tFirst as sunrise or sunset respectively.
twl	data.frame containing twilights and at least tFirst, tSecond and type (alternatively give each parameter separately).
known.coord	a SpatialPoint or matrix object, containing known x and y coordinates (in that order) for the selected measurement period.
plot	logical, if TRUE a plot will be produced.
lnorm.pars	logical, if TRUE shape and scale parameters of the twilight error (log-normal distribution) will be estimated and included in the output (see Details).

Details

Optionally, shape and scale paramters of the twiligth error (in minutes) can be estimated. The error is assumed to follow a log-normal distribution and 0 (elev0) is set 0.1 below the minimum sun elevation angle of estimated twilight times. Those parameters might be of interest for sensitivity analysis or further processing using the R Package SGAT (https://github.com/SWotherspoon/SGAT).

Author(s)

Simeon Lisovski

References

Lisovski, S., Hewson, C.M, Klaassen, R.H.G., Korner-Nievergelt, F., Kristensen, M.W & Hahn, S. (2012) Geolocation by light: Accuracy and precision affected by environmental factors. *Methods in Ecology and Evolution*, DOI: 10.1111/j.2041-210X.2012.00185.x.

Examples

```
data(calib2)
  calib2$tFirst  <- as.POSIXct(calib2$tFirst, tz = "GMT")
  calib2$tSecond <- as.POSIXct(calib2$tSecond, tz = "GMT")
getElevation(calib2, known.coord = c(7.1,46.3), lnorm.pars = TRUE)
```

---

gleTrans *Transformation of *.gle files*

Description

Function to transform *.gle files derived by the software GeoLocator for further analyses in GeoLight.

Usage

```
gleTrans(file)
```

Arguments

file	the full patch and filename with suffix of the *.gle file.

Details

The *.gle file derived by the software "GeoLocator" (Swiss Ornithological Institute) is a table with interpolated light intensities over time gathered from the *.glf file. Furthermore a column defines wether the light intensity passes the defined light intensity threshold in the morning (sunrise) or in the evening (sunset). This information is used in `gleTrans()`, to create a table with two subsequent twilight events (*tFirst, tSecond*) and *type* defining wether *tFirst* refers to sunrise (1) or sunset (2). Date and time information will be transferred into a `GeoLight` appropriate format (see: `as.POSIXct`).

Value

A `data.frame` suitable for further use in `GeoLight`.

Author(s)

Simeon Lisovski

See Also

`glfTrans`

---

glfTrans *Transformation of *.glf files*

Description

Transform *.glf files derived by the software GeoLocator for further analyses in `GeoLight`.

Usage

```
glfTrans(file = "/path/file.glf")
```

Arguments

file	the full patch and filename with suffix of the *.glf file.

Details

The *.glf files produced by the software GeoLocator (Swiss Ornithological Institute) is a table with light intensity measurements over time. `glfTrans` produces a table with these measurements and transfer the data and time information into the format required by `GeoLight` format (see: `as.POSIXct`).

Value

A data.frame suitable for further use in GeoLight.

Author(s)

Simeon Lisovski

See Also

gleTrans; luxTrans for transforming *.lux files produced by *Migrate Technology Ltd*

HillEkstromCalib *Hill-Ekstrom calibration*

Description

Hill-Ekstrom calibration for one or multiple stationary periods.

Usage

```
HillEkstromCalib(tFirst, tSecond, type, twl, site, start.angle = -6,
  distanceFilter = FALSE, distance, plot = TRUE)
```

Arguments

tFirst	vector of sunrise/sunset times (e.g. 2008-12-01 08:30).
tSecond	vector of of sunrise/sunset times (e.g. 2008-12-01 17:30).
type	vector of either 1 or 2, defining tFirst as sunrise or sunset respectively.
twl	data.frame containing twilights and at least tFirst, tSecond and type (alternatively give each parameter separately).
site	a numerical vector assigning each row to a particular period. Stationary periods in numerical order and values >0, migration/movement periods 0
start.angle	a single sun elevation angle. The combined process of checking for minimal variance in resulting latitude, which is the initial value for the sun elevation angle in the iterative process of identifying the latitudes with the least variance
distanceFilter	logical, if TRUE the distanceFilter will be used to filter unrealistic positions
distance	if distanceFilter is set TRUE a threshold distance in km has to be set (see: distanceFilter)
plot	logical, if TRUE the function will give a plot with all relevant information

Details

The *Hill-Ekstrom calibration* has been suggested by Hill & Braun (2001) and Ekstrom (2004), and allows for calibrating data during stationary periods at unknown latitudinal positions. The Hill-Ekstrom calibration bases on an increasing error range in latitudes with an increasing mismatch between light level threshold and the used sun angle. This error is strongly amplified with proximity to the equinox times due to decreasing slope of day length variation with latitude. Furthermore, the sign of the error switches at the equinox, i.e. latitude is overestimated before the equinox and underestimated after the equinox (or vice versa depending on autumnal/vernal equinox, hemisphere, and sign of the mismatch between light level threshold and sun angle). When calculating the positions of a stationary period, the variance in latitude is minimal if the sun elevation angle fits to the defined light level threshold. Moreover, the accuracy of positions increases with decreasing variance in latitudes. **However, the method is only applicable for stationary periods and under stable shading intensities**. The plot produced by the function may help to judge visually if the calculated sun elevation angles are realistic (e.g. site 2 in the example below) or not (e.g. site 3 in the example below.

Value

A `vector` of sun elevation angles corresponding to the Hill-Ekstrom calibration for each defined period.

Author(s)

Simeon Lisovski

References

Ekstrom, P.A. (2004) An advance in geolocation by light. *Memoirs of the National Institute of Polar Research*, Special Issue, **58**, 210-226.

Hill, C. & Braun, M.J. (2001) Geolocation by light level - the next step: Latitude. In: *Electronic Tagging and Tracking in Marine Fisheries* (eds J.R. Sibert & J. Nielsen), pp. 315-330. Kluwer Academic Publishers, The Netherlands.

Lisovski, S., Hewson, C.M, Klaassen, R.H.G., Korner-Nievergelt, F., Kristensen, M.W & Hahn, S. (2012) Geolocation by light: Accuracy and precision affected by environmental factors. *Methods in Ecology and Evolution*, DOI: 10.1111/j.2041-210X.2012.00185.x.

Examples

```
data(hoopoe2)
  hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
  hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
residency <- with(hoopoe2, changeLight(tFirst,tSecond,type, rise.prob=0.1,
                  set.prob=0.1, plot=FALSE, summary=FALSE))
HillEkstromCalib(hoopoe2,site = residency$site)
```

hoopoe1 *Light intensity measurements over time recorded on a migratory bird*

Description

Sunlight intensity measurements over time recorded during the first part of the annual migration of a European Hoopoe (*Upupa epops*). All dates/times are measured in Universal Time Zone (UTC).

Format

A table with 24474 rows and 2 columns, rows corresponding to light measurements recorded in ten-minute intervals (datetime).

Source

Baechler, E., Hahn, S., Schaub, M., Arlettaz, R., Jenni, L., Fox, J.W., Afanasyev, V. & Liechti, F. (2010) Year-Round Tracking of Small Trans-Saharan Migrants Using Light-Level Geolocators. *Plos One*, **5**.

hoopoe2 *Sunrise and sunset times: From light intensity measurement (hoopoe1)*

Description

Sunrise and sunset times derived from light intensity measurements over time (hoopoe1). The light measurements corresponding to the first part of the annual migration of a European Hoopoe (*Upupa epops*).

Format

A table with 340 rows and 3 columns. Each row corresponds to subsequent twilight events ("tFirst" and "tSecond"). The third column ("type") indicates weather the first event is sunrise (1) or sunset (2). All dates/times are measured in Universal Time Zone (UTC).

Source

Baechler, E., Hahn, S., Schaub, M., Arlettaz, R., Jenni, L., Fox, J.W., Afanasyev, V. & Liechti, F. (2010) Year-Round Tracking of Small Trans-Saharan Migrants Using Light-Level Geolocators. *Plos One*, **5**.

Examples

```
data(hoopoe2)
 hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
 hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
coord <- coord(hoopoe2, degElevation=-6)
## plot in a map using package maps
# par(oma=c(5,0,0,0))
# map(xlim=c(-20,40),ylim=c(-10,60),interior=F,col="darkgrey")
# map(xlim=c(-20,40),ylim=c(-10,60),boundary=F,lty=2,col="darkgrey",add=T)
# mtext(c("Longitude (degrees)","Latitude (degrees)"),side=c(1,2),line=c(2.2,2.5),font=3)
# map.axes()
# points(coord,col="brown",cex=0.5,pch=20)
```

---

lightFilter *Filter to remove noise in light intensity measurements during the night*

Description

The filter identifies and removes light intensities oczillating around the baseline or few light intensities resulting in a short light peak during the night. Such noise during the night will increase the calculated twilight events using the function twilightCalc and therewith the manual work to remove these false twilight events.

Usage

```
lightFilter(light, baseline = NULL, iter = 2)
```

Arguments

light	numerical value of the light intensity (usually arbitrary units).
baseline	the light intensity baseline (no light). If Default, it will be calculated as the most frequent value below the mean light intensities.
iter	a numerical value, specifying how many iterations should be computed (see details).

Details

The filter searches for light levels above the baseline and compares the prior and posterior levels. If these values are below the threshold the particular light level will be reduced to the baseline. A few (usually two) iterations might be enough to remove most noise during the night (however, not if such noise occurs at the begining or at the end were not enough prior or posterior values are available).

Value

numerical vector with the new light levels. Same length as the initial light vector.

Author(s)

Simeon Lisovski

Examples

```
night <- rep(0,50); night[runif(4,0,50)] <- 10; night[runif(4,0,50)] <- -5
nightday <- c(night,rep(30,50))
plot(nightday,type="l",ylim=c(-5,30),ylab="light level",xlab="time (time)")
light2 <- lightFilter(nightday, baseline=0, iter=4)
lines(light2,col="red")
legend("bottomright",c("before","after"),lty=c(1,1),col=c("black","red"),bty="n")
```

---

ligTrans *Transformation of *.lig files*

Description

Transform *.lig files derived from geolocator deviced for further analyses in GeoLight.

Usage

```
ligTrans(file)
```

Arguments

file	the full patch and filename with suffix of the *.lux file.

Value

A data.frame suitable for further use in GeoLight.

Author(s)

Tamara Emmenegger

See Also

gleTrans for transforming *.glf files produced by the software GeoLocator (*Swiss Ornithological Institute*)

loessFilter — *Filter to remove outliers in defined twilight times based on smoother function*

Description

This filter defines outliers based on residuals from a local polynomial regression fitting provcess (loess).

Usage

```
loessFilter(tFirst, tSecond, type, twl, k = 3, plot = TRUE)
```

Arguments

tFirst	vector of sunrise/sunset times (e.g. 2008-12-01 08:30).
tSecond	vector of of sunrise/sunset times (e.g. 2008-12-01 17:30).
type	vector of either 1 or 2, defining tFirst as sunrise or sunset respectively.
twl	data.frame containing twilights and at least tFirst, tSecond and type (alternatively give each parameter separately).
k	a measure of how many interquartile ranges to take before saying that a particular twilight event is an outlier
plot	codelogical, if TRUE a plot indicating the filtered times will be produced.

Value

Logical vector matching positions that pass the filter.

Author(s)

Simeon Lisovski & Eldar Rakhimberdiev

luxTrans — *Transformation of *.lux files*

Description

Transform *.lux files derived from *Migrate Technology Ltd* geolocator deviced for further analyses in GeoLight.

Usage

```
luxTrans(file)
```

Arguments

file	the full patch and filename with suffix of the *.lux file.

Details

The *.lux files produced by *Migrate Technology Ltd* are table with light intensity measurements over time. luxTrans produces a table with these measurements and transfer the data and time information into the format required by GeoLight format (see: as.POSIXct).

Value

A data.frame suitable for further use in GeoLight.

Author(s)

Simeon Lisovski

See Also

gleTrans for transforming *.glf files produced by the software GeoLocator (*Swiss Ornithological Institute*)

---

mergeSites *Function to merge sites*

Description

The changeLight functions provides a vector grouping the twilight times into stationary (>0) and movement (0) periods. This function was written to enable the user to merge sites based on the distance between consequtive sites. NOTE: The function requires position estimate and desicison on whether sites should be merged will be made based on the defined distance, the cutoff values and the provided positions. The analysis is this dependent on the accuracy of the position estiamtes and should be applied to positons that were estimated using a sensible sun elevation angle.

Usage

```
mergeSites(tFirst, tSecond, type, twl, site, degElevation,
  distThreshold = 250, alpha = c(0, 15), plot = TRUE)
```

Arguments

tFirst	vector of sunrise/sunset times (e.g. 2008-12-01 08:30).
tSecond	vector of of sunrise/sunset times (e.g. 2008-12-01 17:30).
type	vector of either 1 or 2, defining tFirst as sunrise or sunset respectively.
twl	data.frame containing twilights and at least tFirst, tSecond and type

site	a numerical vector assigning each row to a particular period. Stationary periods in numerical order and values >0, migration/movement periods 0. This vector will be used as the initial state.
degElevation	the sun elevation angle (in degrees) that defines twilight (e.g. -6 for "civil twilight"). Either a single value, a vector with the same length as tFirst or nrow(x).
distThreshold	a numerical value defining the threshold of the distance under which consequtive sites should be merged (in km).
alpha	mean and standard variation for position optimization process.
plot	logical, if TRUE a plot comparing the inital and the final site selection.

Value

A vector with the merged site numbers

Author(s)

Simeon Lisovski

---

refracted *Atmospheric Refraction*

Description

Adjust the solar zenith angle for atmospheric refraction.

Usage

```
refracted(zenith)
```

Arguments

zenith	zenith angle (degrees) to adjust.

Details

Given a vector of solar zeniths computed by zenith, refracted calculates the solar zeniths adjusted for the effect of atmospheric refraction.

unrefracted is the inverse of refracted. Given a (single) solar zenith adjusted for the effect of atmospheric refraction, unrefracted calculates the solar zenith as computed by zenith.

Value

vector of zenith angles (degrees) adjusted for atmospheric refraction.

Examples

```
## Refraction causes the sun to appears higher on the horizon
refracted(85:92)
## unrefracted gives unadjusted zenith (see SGAT)
```

`schedule` *Function for making a data frame summarising residency and movement pattern.*

Description

Function for making a data frame summarising residency and movement pattern.

Usage

```
schedule(tFirst, tSecond, site)
```

Arguments

`tFirst`	date and time of sunrise/sunset (e.g. 2008-12-01 08:30)
`tSecond`	date and time of sunrise/sunset (e.g. 2008-12-01 17:30)
`site`	a `vector`, indicating the residency period of a particular day (see output: `changeLight`)

Value

A `data.frame` with end and start date (yyyy-mm-dd hh:mm, UTC) for each stationary period.

Author(s)

Simeon Lisovski

Examples

```
data(hoopoe2)
  hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
  hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
residency <- changeLight(hoopoe2, rise.prob=0.1, set.prob=0.1, plot=FALSE, summary=FALSE)
schedule(hoopoe2[,1], hoopoe2[,2], site = residency$site)
```

siteMap *Draws sites of residency and adds a convex hull*

Description

Draw a map (from the R Package maps) showing the defined stationary sites

Usage

```
siteMap(crds, site, type = "points", quantiles = c(0.25, 0.75), hull = T,
  map.range = c("EuroAfrica", "AustralAsia", "America", "World"), ...)
```

Arguments

crds	a SpatialPoints or matrix object, containing x and y coordinates (in that order).
site	a numerical vector assigning each row to a particular period. Stationary periods in numerical order and values >0, migration/movement periods 0.
type	either *points*, or *cross* to show all points for each site or only show the mean position of the site with standard deviation.
quantiles	the quantile of the error bars (*cross*) around the median.
hull	logical, if TRUE a convex hull will be plotted around the points of each site.
map.range	some possibilities to choose defined areas ("World (default)", "EuroAfrica","America","AustralAsia").
...	Arguments to be passed to methods, such as graphical parameters (see par).

Details

Standard graphical paramters like pch, cex, lwd, lty and col are implemented. The color can be specified as either a vector of colors (e.g. c("blue", "red", ...)) or as a character string indicating a color ramp (at the moment only "random" and "rainbow" is available)

Author(s)

Simeon Lisovski & Tamara Emmenegger

Examples

```
data(hoopoe2)
 hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
 hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
crds <- coord(hoopoe2, degElevation = -6)
filter <- distanceFilter(hoopoe2, distance = 30)
site <- changeLight(hoopoe2, rise.prob = 0.1, set.prob = 0.1, plot = FALSE,
 summary = FALSE)$site
siteMap(crds[filter,], site[filter], xlim=c(-20,20), ylim=c(0,60),
 lwd=2, pch=20, cex=0.5, main="hoopoe2")
```

solar *Solar Time and Declination*

Description

Calculate solar time, the equation of time and solar declination

Usage

```
solar(tm)
```

Arguments

tm	a vector of POSIXct times.

Details

The solar time, the equation of time and the sine and cosine of the solar declination are calculted for the times specified by `tm` using the same methods as www.esrl.noaa.gov/gmd/grad/solcalc/.

Value

A list containing the following vectors.

solarTime	the solar time (degrees)
eqnTime	the equation of time (minutes of time)
sinSolarDec	sine of the solar declination
cosSolarDec	cosine of the solar declination

See Also

zenith

Examples

```
## Current solar time
solar(Sys.time())
```

staroddiTrans *Transformation of staroddi files*

Description

Transform staroddi files derived from geolocator deviced for further analyses in GeoLight.

Usage

```
staroddiTrans(file)
```

Arguments

file	the full patch and filename of the staroddi file.

Value

A data.frame suitable for further use in GeoLight.

Author(s)

Tamara Emmenegger

trip2kml *Write a file which plots a trip in Google Earth*

Description

This function creates a .kml file from light intensity measurements over time that can ve viewed as a trip in Google Earth.

Usage

```
trip2kml(file, tFirst, tSecond, type, degElevation,
  col.scheme = "heat.colors", point.alpha = 0.7, cex = 1,
  line.col = "goldenrod")
```

Arguments

file	A character expression giving the whole path and the name of the resulting output file including the .kml extension.
tFirst	date and time of sunrise/sunset (e.g. 2008-12-01 08:30)
tSecond	date and time of sunrise/sunset (e.g. 2008-12-01 17:30)
type	either 1 or 2, defining tFirst as sunrise or sunset respectively

degElevation	sun elevation angle in degrees (e.g. -6 for "civil twilight"). Either a single value, a vector with the same length as tFirst.
col.scheme	the color scheme used for the points. Possible color schemes are: rainbow, heat.colors, topo.colors, terrain.colors.
point.alpha	a numerical value indicating the transparency of the point colors on a scale from 0 (transparent) to 1 (opaque).
cex	numerical value for the size of the points.
line.col	An character expression (any of colors or hexadecimal notation), or numeric indicating the color of the line connecting the point locations.

Value

This function returns no data. It creates a .kml file in the in the defined path.

Author(s)

Simeon Lisovski and Michael U. Kemp

Examples

```
data(hoopoe2)
  hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
  hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
filter <- distanceFilter(hoopoe2,distance=30)
## takes time
## trip2kml("trip.kml", hoopoe2$tFirst[filter], hoopoe2$tSecond[filter], hoopoe2$type[filter],
## degElevation=-6, col.scheme="heat.colors", cex=0.7,
## line.col="goldenrod")
```

---

tripMap *Draw the positions and the trip on a map*

Description

Draw a map (from the R Package maps) with calculated positions connected by a line

Usage

```
tripMap(crds, equinox = TRUE, map.range = c("EuroAfrica", "AustralAsia",
  "America", "World"), legend = TRUE, ...)
```

Arguments

crds	a SpatialPoints or matrix object, containing x and y coordinates (in that order).
equinox	logical; if TRUE, the equinox period(s) is shown as a broken blue line.
map.range	some possibilities to choose defined areas (default: "World").
legend	logical; if TRUE, a legend will be added to the plot.
...	Arguments to be passed to methods, such as graphical parameters (see par).

Author(s)

Simeon Lisovski

Examples

```
data(hoopoe2)
 hoopoe2$tFirst <- as.POSIXct(hoopoe2$tFirst, tz = "GMT")
 hoopoe2$tSecond <- as.POSIXct(hoopoe2$tSecond, tz = "GMT")
crds <- coord(hoopoe2, degElevation = -6)
tripMap(crds, xlim = c(-20,20), ylim = c(0,60), main="hoopoe2")
```

---

trnTrans *Transformation of *.trn files*

Description

Transform *.trn files derived from geolocator deviced for further analyses in GeoLight.

Usage

```
trnTrans(file)
```

Arguments

file	the full patch and filename of the *.trn file.

Value

A data.frame suitable for further use in GeoLight.

Author(s)

Tamara Emmenegger

`twilight` *Times of Sunrise and Sunset*

Description

Estimate time of sunrsie or sunset for a given day and location

Usage

```
twilight(tm, lon, lat, rise, zenith = 96, iters = 3)
```

Arguments

`tm`	vector of approximate times of twilight.
`lon`	vector of longitudes.
`lat`	vector of latitudes.
`rise`	logical vector indicating whether to compute rise or set.
`zenith`	the solar zenith angle that defines twilight.
`iters`	number of iteratve refinements made to the initial approximation.

Details

`twilight` uses an iterative algorithm to estimate times of sunrise and sunset.

Note that these functions return the twilight that occurs on the same date GMT as `tm`, and so sunset may occur before sunrise, depending upon latitude.

Solar declination and equation of time vary slowly over the day, and so the values of the Solar declination and equation of time at sunrise/sunset are well approximated by their values at 6AM/6PM local time. The sun's hour angle and hence sunrise/sunset for the required zenith can then be caclulates from these approximations. The calculation is then repeated using the approximate sunrise/sunset times to derive more accurate values of the Solar declination and equation of time and hence better approximations of sunrise/sunset. The process is repreated and is accurate to less than 2 seconds within 2 or 3 iterations.

`sunrise` and `sunset` are simple wrappers for `twilight`.

Value

a vector of twilight times.

`twilightCalc` *Calculate twilight events (sunrise/sunset) from light intensity measurements over time*

Description

Defines twilight events (sunrise/sunset) at times when the light intensity measurements (*light*) pass the defined light intensity threshold. An interactive plot can be drawn to assess the calculations and improve e.g. select only the realistic events.

Usage

```
twilightCalc(datetime, light, LightThreshold = TRUE, preSelection = TRUE,
  maxLight = NULL, ask = TRUE, nsee = 500, allTwilights = FALSE)
```

Arguments

`datetime`	date and time of light intensity measurements e.g. 2008-12-01 08:30 "UTC" (see: `as.POSIXct`,time zones).
`light`	`numerical` value of the light intensity (usually arbitrary units).
`LightThreshold`	the light intensity threshold for the twilight event calibration. If `Default`, it will be set slightly above (3 units) the baseline level (measurement during the night).
`preSelection`	codelogical, if TRUE a pre selection of all calculated twiligth events will be offered within the interactive process (ask=TRUE).
`maxLight`	if the geolocator record the maximum light value of a certain time span, give the interval of maximum recordings in minutes (e.g. 5).
`ask`	`logical`, if TRUE the interactive plot will start after the calculation.
`nsee`	number of points to plot per screen.
`allTwilights`	`logical`, if TRUE the function returns a list with two tables

Value

if allTwilights=FALSE, a `data frame`. Each row contains two subsequent twilight events (*tFirst, tSecond*) and *type* defining wether *tFirst* refers to sunrise (1) or sunset (2). If allTwilights=TRUE, a `list` with the data frame described in the previous sentence and a data frame with all light intensities and a column describing whether each row refers to sunrise (1), sunset (2) or to none of these categories (0).

Note

Depending on shading during light intensity measurements (e.g. due to vegetation, weather, etc., see Lisovski et *al.* 2012) the light intensities may pass the light intensity threshold several times during the day, resulting false sunrises and sunsets. It is highly recommended to check the derived events visually (`ask=TRUE`).Twilight events can be deleted and undeleted by clicking the (first) mouse button at the particular position in the graph. The second mouse buttom (or esc) moves the time series forward. Note, that a backward option is not included.

Author(s)

Simeon Lisovski

`zenith` *Solar Zenith Angle*

Description

Calculate the solar zenith angle for given times and locations

Usage

```
zenith(sun, lon, lat)
```

Arguments

`sun`	list of solar time and declination computed by `solar`.
`lon`	vector of longitudes.
`lat`	vector latitudes.

Details

`zenith` uses the solar time and declination calculated by `solar` to compute the solar zenith angle for given times and locations, using the same methods as www.esrl.noaa.gov/gmd/grad/solcalc/. This function does not adjust for atmospheric refraction see `refracted`.

Value

A vector of solar zenith angles (degrees) for the given locations and times.

See Also

`solar`

Examples

```
## Approx location of Sydney Harbour Bridge
lon <- 151.211
lat <- -33.852
## Solar zenith angle for noon on the first of May 2000
## at the Sydney Harbour Bridge
s <- solar(as.POSIXct("2000-05-01 12:00:00","EST"))
zenith(s,lon,lat)
```

Index