Reference manual for the trackit package documentation

of

March 16, 2009

Version 0.2-57

Date 2009-03-16

Title Track tagged individuals from light measurements

Author Anders Nielsen <anders.nielsen@hawaii.edu>, John Sibert <sibert@hawaii.edu>

Maintainer Anders Nielsen <anders.nielsen@hawaii.edu>

Depends R (>= 2.2.0), MASS, date

Suggests

Description Track tagged individuals from raw light measurements. This package contain tools for estimating the track from a time series of light

License BSD

URL https://www.soest.hawaii.edu/tag-data/software/

R topics documented:

`angle2light` *angle2light*

Description

A dataset describing the estimated relationship between solar altitude in degrees (-90)-(90) above the horizon and the transformed light level measured in the tag

Usage

```
data(angle2light)
```

Format

A data.frame containing two columns `angle` and `light`. There are 1801 observations (rows).

Details

This data set, or another one of similar structure is required input for the function `light.simulator`.

This particular data set is extracted from the fitted relationship between altitude and light in the `mooring` example.

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

Examples

```
data(angle2light)
plot(angle2light, type='l', lwd=3, col='red')
```

`deltat` *deltat*

Description

A table of DeltaT, which is the empirical difference between dynamic time (DT) and universal time UT in seconds.

Usage

```
data(deltat)
```

Format

A data frame with 70 observations on the following 5 variables.

`day` a numeric vector

`month` a numeric vector

`year` a numeric vector

`DTminusUT` a numeric vector

`JDE` a numeric vector

Details

A small internal detail is that if the dataset is loaded as below the column JDE will not show up, but if it is not loaded it is available for internal use.

Source

http://www.stjarnhimlen.se/comp/time.html

Examples

```
data(deltat)
plot(deltat[,3]+deltat[,2]/12, deltat[,4], xlab="year", ylab=expression(Delta*T))
```

`drifter` *drifter*

Description

Data from a Wildlife Computer pop-up archival tag (PAT, version 2)

Usage

```
data(drifter)
```

Format

A data frame with 4536 observations on the following 9 variables.

- **`year`** a numeric vector
- **`month`** a numeric vector
- **`day`** a numeric vector
- **`hour`** a numeric vector
- **`min`** a numeric vector
- **`sec`** a numeric vector
- **`depth`** a numeric vector
- **`light`** a numeric vector
- **`temp`** a numeric vector

Details

Notice in the example that `scan=FALSE`. This is the correct option, because PAT version 2 scans the data and saves only the data around the solar events.

Source

Data provided by Mike Musyl.

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

Examples

```
data(drifter)
prep.track<-prepit(drifter, fix.first=c(360-161.45,22.85,2002,9,10,0,0,0),
                            fix.last=c(360-159.87,21.95,2003,5,21,0,0,0), scan=FALSE)
# try to run (not in the example, because it takes a while)
#   fit<-trackit(prep.track)
#   plot(fit)
```

fitmap *Plot the most probable track on a map*

Description

Plot the most probable track and possibly its confidence region on a map

Usage

```
fitmap(x, ci=FALSE, ...)
```

Arguments

`x`	An object returned from the `trackit` function
`ci`	If TRUE the 95% confidence region is added to the plot
`...`	additionel arguments to `plotbasemap`

Value

Value is `NULL`

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩

get.avhrr.sst *Get SST-field from avhrr source*

Description

This function allows easy access to a avhrr SST database.

Usage

```
get.avhrr.sst(track, lonlow, lonhigh, latlow, lathigh, folder = tempdir(),
              server = "http://coastwatch.pfeg.noaa.gov/coastwatch/CWBrowserWW360.
              product = "TAGssta", nday = "8day", centertime="12")
```

Arguments

`track` A single `data.frame` containing a track or a `list` of `data.frames` each containing a track. The idea is that the function should only download the SST-data spanning the region and period of the tracks that needs to be analyzed.

`lonlow` The lowest longitude where SST is downloaded

`lonhigh` The highest longitude where SST is downloaded

`latlow` The lowest latitude where SST is downloaded

`lathigh` The highest latitude where SST is downloaded

`folder` Is where the downloaded raw data files are stored. This defaults to a temporary directory.

`server` The url of the server

`product` The 7-character name of the imagery product. Refer to the Coastwatch site for the complete list of relevant SST products. `http://coastwatch.pfeg.noaa.gov/coastwatch/CWBrowserWW360.jsp?get=gridData\&dataSet=`

`nday` Time resolution should be either '5day' or '8day'

`centertime` The time stamp of the image '00' is from midnight to midnight, and '12' is from noon to noon.

Details

Value

The path returned from the function is where all the raw SST files are saved.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩, Chi Lam ⟨chihinl@usc.edu⟩, Dave Foley ⟨Dave.Foley@noaa.gov⟩

References

TALK TO DAVE FOLEY

See Also

`get.sst.from.server`, `get.avhrr.sst`

Examples

```
# No example supplied here
```

`get.blended.sst` *Get SST-field from blended source*

Description

This function allows easy access to a blended SST database.

Usage

```
get.blended.sst(track, lonlow, lonhigh, latlow, lathigh, folder = tempdir(),
  server = "http://coastwatch.pfeg.noaa.gov/coastwatch/CWBrowserWW360.jsp?get=gridD
  nday = "5day")
```

Arguments

`track` A single `data.frame` containing a track or a `list` of `data.frames` each containing a track. The idea is that the function should only download the SST-data spanning the region and period of the tracks that needs to be analyzed.

`lonlow` The lowest longitude where SST is downloaded

`lonhigh` The highest longitude where SST is downloaded

`latlow` The lowest latitude where SST is downloaded

`lathigh` The highest latitude where SST is downloaded

`folder` Is where the downloaded raw data files are stored. This defaults to a temporary directory.

`server` the url of the server

`nday` Time resolution should be either '5day' or '8day'

Details

Value

The path returned from the function is where all the raw SST files are saved.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩, Chi Lam ⟨chihinl@usc.edu⟩, Dave Foley ⟨Dave.Foley@noaa.gov⟩

References

TALK TO DAVE FOLEY

See Also

`get.sst.from.server`, `get.avhrr.sst`

Examples

```
# No example supplied here
```

---

get.sst.from.server

Get SST-field from server

Description

This function allows easy access to three different SST sources that has been setup for this purpose. Data is downloaded from within R and stored in a format ready to be used.

Usage

```
get.sst.from.server(track, lonlow, lonhigh, latlow, lathigh, folder = tempdir(),
       server = "http://atlas.nmfs.hawaii.edu/cgi-bin/reynolds_extract.py")
```

Arguments

`track` A single `data.frame` containing a track or a `list` of `data.frame`s each containing a track. The idea is that the function should only download the SST-data spanning the region and period of the tracks that needs to be analyzed.

`lonlow` The lowest longitude where SST is downloaded

`lonhigh` The highest longitude where SST is downloaded

`latlow` The lowest latitude where SST is downloaded

`lathigh` The highest latitude where SST is downloaded

`folder` Is where the downloaded raw data files are stored. This defaults to a temporary directory.

`server` Presently three servers are available. The default is the fairly coarse Reynold's one 1x1-degree 8-day composites. This source is fast to download, but may be too coarse in some areas. The two other servers are the AVHRR-GAC 3-day and AVHRR-GAC 8-day composites these have a 0.1x0.1-degree resolution. The server names are:

`http://atlas.nmfs.hawaii.edu/cgi-bin/gac3day_extract.py`

and

`http://atlas.nmfs.hawaii.edu/cgi-bin/gac8day_extract.py`

Details

The servers has been set up to extract SST-fields that covers a track (or a set of tracks) in simple way from within R. To use the default source type a command similar to:

sst.path <- get.sst.from.server(track1)

Notice 'track1' can be replaced by a list of tracks like:

sst.path <- get.sst.from.server(list(track1, track2)) to obtain an SST-field covering a set of tracks.

To use one of the two other servers simply supply the server name as in:

```
sst.path <- get.sst.from.server(list(track1, track2), server='http://atlas.nmfs.haw
bin/gac3day_extract.py')
```

or

```
sst.path <- get.sst.from.server(list(track1, track2), server='http://atlas.nmfs.haw
bin/gac8day_extract.py')
```

To use a user supplied SST-source please see documentation for the function ....

Value

The path returned from the function is where all the raw SST files are saved.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩, Russell Moffitt ⟨Russell.Moffitt@noaa.gov⟩

References

TALK TO RUSS

See Also

`get.blended.sst`

Examples

```
  # No example supplied here
```

gmt3 *gmt3*

Description

A dataset of land polygons used for plotting maps

Usage

```
data(gmt3)
```

Format

A data frame with 504522 observations on the following 2 variables.

`longitude` a numeric vector

`latitude` a numeric vector

Source

http://gmt.soest.hawaii.edu/

Examples

```
data(gmt3)
```

light.simulator *Simulate artificial datasets*

Description

This function is mainly used for model validation by the package authors. For a given set of model parameters it simulate a dataset that follows the model. It can be useful for setting up simulation studies.

Usage

```
light.simulator(u = 0, v = 0, D = 100, ss1 = 80, ss2 = 15, ss3 = 2.5, rho = 0.05, b
```

Arguments

`u` Model parameter (see reference)

`v` Model parameter (see reference)

`D` Model parameter (see reference)

`ss1` Model parameter (see reference)

`ss2` Model parameter (see reference)

`ss3` Model parameter (see reference)

`rho` Model parameter (see reference)

`bsst` Model parameter (see reference)

`sssst` Model parameter (see reference)

`rad` Model parameter (see reference)

`sundata` A dataset like the built-in `angle2light` describing the relationship between solar altitude angle and light level

`sstdates` Dates where sstdata are to be used

`fix.first` Release position

`t` Light sampling times (before scanning)

`window` Two numbers determining how large an interval around each solar event is used. First the mid-point is selected as the point that best separates day from night (details in reference), and then two numbers are used in the following way: The first number is the fraction (of a 24-hour period) from the mid-point towards the day side of the solar event. The second number is the fraction towards the night side. So the default `c(0.05, 0.01)` corresponds to using an interval that 14.4 minutes towards the night side and 72 minutes towards the day side. 86.4 minutes total.

`tmpfile` Name of temporary file

`datfile` Name of data file

`truefile` Name of file containing the true simulated track

`keepfiles` Logical vector determining what files to keep

`localsstfolder` Option to supply saved or own sst files

`sst.fun` supply your function to retrive sst

`from.ystr` Is an integer vector with two elements describing what part of the file name describe the year of the first date the data file represents. For instance if the names of the data files all have the format `RSyyyyddd_YYYYDDD.dat`, where `yyyy` is the year of the first date the argument should be `c(3,6)`.

`from.dstr` Is an integer vector with two elements describing what part of the file name describe the 'number of days into the year' of the first date the data file represents.

`to.ystr` Is similar to `from.ystr`, but here for the year of the last date the data file represents.

`to.dstr` Is similar to `from.dstr`, but here for the 'number of days into the year' of the last date the data file represents.

Value

A list containing the needed information about the simulation

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

See Also

`trackit`

Examples

```
# No examples provided here, but try the ones in ?drifter and ?mooring
```

mooring *mooring*

Description

Data from a Wildlife Computers archival tag (MK9, version 1.03)

Usage

```
data(mooring)
```

Format

A data frame with 323418 observations on the following 9 variables.

`year` a numeric vector

`month` a numeric vector

`day` a numeric vector

`hour` a numeric vector

`min` a numeric vector

`sec` a numeric vector

`depth` a numeric vector

`light` a numeric vector

`temp` a numeric vector

Source

Data is provided by Bruno Leroy from the Oceanic Fisheries Programme of the Secretariat of the Pacific Community.

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

Examples

```
data(mooring)
prep.track<-prepit(mooring, fix.first=c(166.47,-22.48,2003,9,8,2,32,0),
                            fix.last=c(166.47,-22.48,2004,4,19,16,49,0))
# try to run (not in the example, because it takes a while)
#   fit<-trackit(prep.track)
#   plot(fit)
```

plotbasemap *Plot land area on a map with colored polygons*

Description

Plots a map within given rectangular region showing land areas as colored polygons. Requires the mapping utility GMT.

Usage

```
plotbasemap(lon1, lon2, lat1, lat2, grid=FALSE, zoom=FALSE,
        landcolor="darkgreen", seacolor="lightblue", data=gmt3)
```

Arguments

`lon1`	Longitude of lower left corner of rectangle
`lon2`	Longitude of upper right corner of rectangle
`lat1`	Latitude of lower left corner of rectangle
`lat2`	Latitude of upper right corner of rectangle
`grid`	Whether to plot grid lines on map
`zoom`	Whether to start in interactive zoom mode
`landcolor`	Color of polygons
`seacolor`	Color of ocean
`data`	dataset to use

Details

A map is plotted with polygons clipped at borders of map region.

If the function is started in zoom mode two left-clicks on the map will zoom it to the rectangle spanned by the two points. This zooming is repeated until a right-click on the map is done.

Value

Value is `NULL`

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩, and Pierre Kleiber.

Examples

```
plotbasemap(8,13,53,58)
```

`plotlat` *Latitude plot*

Description

Nice plot of the latitude component of the most probable track.

Usage

```
plotlat(fit, exr = NULL, mid=FALSE)
```

Arguments

`fit`	The fitted object as returned from the function `trackit`
`exr`	A vector of points. The y-range of the plot is extended to also span these points (if necessary).
`mid`	Mainly used internally. If `TRUE` it is assumed that this plot is placed between a similar longitude and SST plot, and the first axis is omitted.

Value

No value is returned. This function is invoked for its side effects.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

See Also

`trackit`

Examples

```
# No examples provided here, but try the ones in ?drifter and ?mooring
```

plotlon *Longitude plot*

Description

Nice plot of the longitude component of the most probable track.

Usage

```
plotlon(fit, exr = NULL, top = FALSE)
```

Arguments

`fit`	The fitted object as returned from the function `trackit`
`exr`	A vector of points. The y-range of the plot is extended to also span these points (if necessary).
`top`	Mainly used internally. If `TRUE` it is assumed that this plot is placed on top of a similar latitude plot, and the axis are adjusted accordingly.

Value

No value is returned. This function is invoked for its side effects.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

See Also

`trackit`

Examples

```
# No examples provided here, but try the ones in ?drifter and ?mooring
```

`plotsst` *SST plot*

Description

Nice plot of the SST component of the most probable track.

Usage

```
plotsst(fit, exr = NULL)
```

Arguments

`fit`	The fitted object as returned from the function `trackit`
`exr`	A vector of points. The y-range of the plot is extended to also span these points (if necessary).

Value

No value is returned. This function is invoked for its side effects.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (in press).

See Also

`trackit`

Examples

```
  # No examples provided here, but try the ones in ?drifter and ?mooring
```

`plot.trackit` *Plot a fitted track*

Description

Plots a nice representation of the fitted most probable track.

Usage

```
plot.trackit(x, onlylonlat=FALSE, ...)
```

Arguments

`x`	The fitted object as returned from the function `trackit`
`onlylonlat`	For fitted objects with sst this option turns off the SST panel in the plot
`...`	additional graphical parameters (currently not used)

Value

No value is returned. This function is invoked for its side effects.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

See Also

`trackit`

Examples

```
# No examples provided here, but try the ones in ?drifter and ?mooring
```

prepit *Prepare a track for light based geolocation*

Description

This function prepares the track for light based geolocation, by converting it to the right format, pre-computing various variables, and selecting the intervals around sunrise and sunset where the data is used in the geolocation model (if needed).

Usage

```
prepit(track, fix.first, fix.last, scan = TRUE, window = c(0.05, 0.01),
       tmpfile = "input.dat", datfile = "ukf.dat",
       keepfiles = c(FALSE, FALSE), internal = TRUE, sst=NULL,
       from.ystr=c(3,6), from.dstr=c(7,9), to.ystr=c(11,14), to.dstr=c(15,17),
       localsstfolder=NULL)
```

Arguments

track
: A data frame with the columns `year month day hour min sec depth light temp`. The temperature is not yet used by the model, so just put zeroes in that column.

fix.first
: A vector of eight numbers giving the (longitude, latitude, year, month, day, hour, minute, second) of the release position. Longitude in degrees east, and time as Greenwich Mean Time (GMT).

fix.last
: A vector of eight numbers giving the (longitude, latitude, year, month, day, hour, minute, second) of the recapture position. Longitude in degrees east, and time as Greenwich Mean Time (GMT).

scan
: This can be either `TRUE` (default) or `FALSE`. If `TRUE` a scanning algorithm (described in the reference) is applied to locate the influential parts of the data series, which are the intervals around sunrise and sunsets. If `FALSE` it is assumed that data only consists of data near sunrise and sunsets, which is for instance true if they come from a PAT tag.

window
: If `scan=TRUE` these two numbers determines how large an interval around each solar event is used. First the mid-point is selected as the point that best separates day from night (details in reference), and then two numbers are used in the following way: The first number is the fraction (of a 24-hour period) from the mid-point towards the day side of the solar event. The second number is the fraction towards the night side. So the default `c(0.05, 0.01)` corresponds to using an interval that 14.4 minutes towards the night side and 72 minutes towards the day side. 86.4 minutes total.

tmpfile
: A string specifying the name of a temporary file.

datfile
: A string specifying the name of the file to be used by the function `trackit`

`keepfiles` A logical vector of two elements. First indicates if the temporary file is to be saved. Second indicates if the data file for `trackit` is to be saved. The default is `FALSE` in both cases, which means that everything needed is stored in the object returned from this function.

`internal` Logical indicating if the scanned track is to be stored in the returned object (default). If the track is long it may be advisable to turn this off in combination with `keepfile=c(FALSE,TRUE)`.

`sst` A matrix containing columns year month day hour min sec sst

`from.ystr` Is an integer vector with two elements describing what part of the file name describe the year of the first date the data file represents. For instance if the names of the data files all have the format `RSyyyyddd_YYYYDDD.dat`, where `yyyy` is the year of the first date the argument should be `c(3,6)`.

`from.dstr` Is an integer vector with two elements describing what part of the file name describe the 'number of days into the year' of the first date the data file represents.

`to.ystr` Is similar to `from.ystr`, but here for the year of the last date the data file represents.

`to.dstr` Is similar to `from.dstr`, but here for the 'number of days into the year' of the last date the data file represents.

`localsstfolder`

If the SST source is a bunch of files in a local folder this is where the folder name is given as a string

Details

See reference.

Value

A list containing all data needed in the model `trackit`, or references to files containing the data.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (submitted).

See Also

`trackit`

Examples

```
# No examples provided here, but try the ones in ?drifter and ?mooring
```

print.trackit *Print trackit object*

Description

Prints a pretty summary of an object of class `trackit`

Usage

```
print.trackit(x, ...)
```

Arguments

`x`	an object of class `trackit` typically generated with the `trackit` function.
`...`	additional arguments

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩, John Sibert ⟨sibert@hawaii.edu⟩

See Also

`trackit`

scroll *Scrolls through the light record*

Description

Scrolls through the light record

Usage

```
scroll(track, n=1000)
```

Arguments

`track`	A data.frame with a light column
`n`	An integer step size

Value

No value is returned. This function is invoked for its side effects.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

trackit-internal *Internal trackit objects*

Description

Internal trackit objects.

Details

These are not to be called by the user.

trackit *trackit: Light based tracking*

Description

This function fits a state space model developed to estimate the most probable track of archival tagged marine animals. The underlying movement model is a random walk with drift, and the observations are the light measurements surrounding each solar event.

Usage

```
trackit(prep.track, a2lpoints=15,
        u.init=0, v.init=0, D.init=100,
        ss1.init=1, ss2.init=5, ss3.init=1, rho.init=0.01,
        bsst.init=0, sssst.init=0.01, rad.init=200, dep1.init=0, dep2.init=0,
        phi.init=c(60,rep((200-60)/(a2lpoints-1),a2lpoints-1)),
        init=c(u.init,v.init,D.init,ss1.init,ss2.init,ss3.init,rho.init,
               bsst.init,sssst.init,rad.init,dep1.init,dep2.init,phi.init),
        u.ph=-1, v.ph=-1, D.ph=3, ss1.ph=2, ss2.ph=2, ss3.ph=2, rho.ph=3,
        bsst.ph=-1, sssst.ph=2, rad.ph=3, dep1.ph=-1, dep2.ph=-1,phi.ph=1,
        phase=c(u.ph,v.ph,D.ph,ss1.ph,ss2.ph,ss3.ph,rho.ph,
                bsst.ph,sssst.ph,rad.ph,dep1.ph,dep2.ph,phi.ph),
        blue.light.only=FALSE, save.dir=NULL)
```

Arguments

`prep.track` A prepared track, which is the object returned from the function `prepit`

`a2lpoints` The number of support points to be used in the cubic spline approximation of the relationship between solar altitude angle and the light reading in the tag.

`u.init` Set initial value for model parameter u

`v.init` Set initial value for model parameter v

`D.init` Set initial value for model parameter D

`ss1.init` Set initial value for model parameter σ_1^2

`ss2.init` Set initial value for model parameter σ_2^2

`ss3.init` Set initial value for model parameter σ_3^2

`rho.init` Set initial value for model parameter ρ

`bsst.init` Set initial value for model parameter b_{sst}

`sssst.init` Set initial value for model parameter σ_{sst}^2

`rad.init` Set initial value for model parameter r (the radius)

`dep1.init` Set initial value for model parameter d_1

`dep2.init` Set initial value for model parameter d_2

`phi.init` Set initial value for model parameter ϕ (a vector of a2lpoints elements)

`init` A vector of initial values for all model parameters. Sticking to the model description in the reference the order of the elements are: u, v, D, σ_1^2, σ_2^2, σ_3^2, ρ, b_{sst}, σ_{sst}^2, r, d_1, d_2, ϕ_1, ..., ϕ_n.

`u.ph` Set estimation phase for model parameter u. The options are: '-1' (kept fixed at initial value), '1' (first phase reserved for phi) , '2' (second phase), or '3' (third phase). If estimation of u is turned on it is recommended to use phase '3'.

`v.ph` Set estimation phase for model parameter v. If estimation of v is turned on it is recommended to use phase '3'.

`D.ph` Set estimation phase for model parameter D

`ss1.ph` Set estimation phase for model parameter σ_1^2

`ss2.ph` Set estimation phase for model parameter σ_2^2

`ss3.ph` Set estimation phase for model parameter σ_3^2

`rho.ph` Set estimation phase for model parameter ρ

`bsst.ph` Set estimation phase for model parameter b_{sst}. If estimation of b_{sst} is turned on it is recommended to use phase '2'.

`sssst.ph` Set estimation phase for model parameter σ_{sst}^2

`rad.ph` Set estimation phase for model parameter r (the radius)

`dep1.ph` Set estimation phase for model parameter d_1

`dep2.ph` Set estimation phase for model parameter d_2

`phi.ph` Set one common estimation phase for model parameters in ϕ

`phase` A vector to set the phase in which each parameter is to be optimized. It is only recommended to tinker with these it you know what you are doing. There are three phases 1, 2, and 3. If it is desired to keep a parameter fixed at its initial value the phase should be set to -1. The order is the same as for the initial values, except that the phase of all the ϕ parameters are set by one element (the last one).

`blue.light.only`
A logic. If TRUE only solar angles between -3 and 5 are used. This option is under development and should not be used yet. Most often there is no convergence and if there is it should not be trusted.

`save.dir` Optionally a string specifying a permanent directory name

Details

See reference.

Value

A list containing all relevant information about the model fit including the estimated most probable track.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

Nielsen, A., and Sibert, J.R. 2007. State space model for light based tracking of marine animals. Can. J. Fish. Aquat. Sci. (in press).

See Also

`prepit`

Examples

```
  # No examples provided here, but try the ones in ?drifter and ?mooring
```

`two.layer.depth.corr` *Two layer depth correction*

Description

Two layer depth correction done by estimating two extinction coefficients, As suggested be Phil Ekstrom

Usage

```
two.layer.depth.corr(track, daybyday=FALSE, D0=50)
```

Arguments

`track`	A data.frame with a light column and corresponding depth, day month and year columns
`daybyday`	If TRUE the this is done separately for each days data with separate coefficients for each day
`D0`	The depth that separates the two layers

Value

A data.frame similar to the input, but with the light column corrected.

Author(s)

Anders Nielsen ⟨anders.nielsen@hawaii.edu⟩ and John Sibert ⟨sibert@hawaii.edu⟩

References

`http://www.lotek.com/irradiance.pdf` (2002 paper by Phil Ekstrom)

Index