Package: activity (via r-universe)

October 21, 2024

Type Package

Title Animal Activity Statistics

Version 1.3.4

Author Marcus Rowcliffe

Maintainer Marcus Rowcliffe <marcus.rowcliffe@ioz.ac.uk>

Description Provides functions to express clock time data relative to anchor points (typically solar); fit kernel density functions to animal activity time data; plot activity distributions; quantify overall levels of activity; statistically compare activity metrics through bootstrapping; evaluate variation in linear variables with time (or other circular variables).

License GPL-3

Encoding UTF-8

Depends methods

Imports pbapply

LazyData true

RoxygenNote 7.2.3

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

RemoteUrl https://github.com/marcusrowcliffe/activity

RemoteRef HEAD

RemoteSha 8ef1a81d92c60cbd4546f24a6f269dae649b1ab1

Contents

`activity` *Animal activity statistics*

Description

Provides functions to estimate and compare activity parameters from sensor data.

Details

Sensors that record active animals (eg camera traps) build up a record of the distribution of activity over the course of the day. Records are more frequent when animals are more active, and less frequent or absent when animals are inactive. The area under the distribution of records thus contains information on the overall level of activity in a sampled population. This package provides tools for plotting activity distributions, quantifying the overall level of activity with error, and statistically comparing distributions through bootstrapping.

The core function is `fitact`, which creates an `actmod` object containing the circular kernel PDF, and the activity level estimate derived from this. The generic plot function for `actmod` objects plots the distribution. Functions starting with `compare` make statistical comparisons between distributions or activity estimates. Note that all time or other circular data should be in radians (in the range 0 to 2*pi).

References

Rowcliffe, M., Kays, R., Kranstauber, B., Carbone, C., Jansen, P.A. (2014) Quantifying animal activity level using camera trap data. Methods in Ecology and Evolution.

actmod-class *Activity model class.*

Description

An S4 class describing activity models fitted to time of observation data.

Slots

`data` Object of class `"numeric"`, the input data.

`wt` Object of class `"numeric"`, weights applied to the data.

`bw` Object of class `"numeric"`, kernel bandwidth.

`adj` Object of class `"numeric"`, kernel bandwidth adjustment multiplier.

`pdf` Object of class `"matrix"` describing fitted probability density function: Column 1: A regular sequence of radian times at which PDF evaluated; range is [0, 2*pi] if unbounded, and sequence steps are range difference divided by 512. Column 2: Corresponding circular kernel PDF values. Additionally if errors bootstrapped: Column 3: PDF standard error. Column 4: PDF lower 95% confidence limit. Column 5: PDF upper 95% confidence limit.

`act` Object of class `"numeric"` giving activity level estimate and, if errors boostrapped, standard error and 95 percent confidence limits.

BCIspeed *Animal speed data*

Description

Barro Colorado Island 2008 data: speeds of animal passages past camera traps (`speed`), together with species (`species`) and time of day (`time`) for each record.

Format

A dataframe with 2204 observations and 3 variables.

Source

http://dx.doi.org/10.6084/m9.figshare.1160536

BCItime *Animal record time of day data*

Description

Barro Colorado Island 2008 data: times of day at which animal records occured (`time`), together with species (`species`).

Format

A dataframe with 17820 observations and 2 variables.

Source

http://dx.doi.org/10.6084/m9.figshare.1160536

bwcalc *Calculate circular kernel bandwidth.*

Description

Uses an optimisation procedure to calculate the circular kernel bandwidth giving the best fit to the data.

Usage

```
bwcalc(dat, K = 3)
```

Arguments

`dat`	Numeric data vector of radian times.
`K`	Integer number of values of kappa over which to maximise (see references for details).

Details

Mainly for internal use.

Value

Single numeric bandwidth value.

References

Ridout, M.S. & Linkie, M. (2009) Estimating overlap of daily activity patterns from camera trap data. Journal of Agricultural Biological and Environmental Statistics, 14, 322-337.

cmean *Circular mean*

Description

Calculates the average direction of a set of radian circular values.

Usage

```
cmean(x, ...)
```

Arguments

x	A vector of radian values.
...	Arguments passed to mean.

Details

The base::mean function is use internally, and additional arguments, e.g for missing data handling, are passed to this.

Value

A radian value giving mean direction.

See Also

mean

Examples

```
data(BCItime)
times <- subset(BCItime, species=="ocelot")$time*2*pi
cmean(times)
```

compareAct *Compare activity level estimates*

Description

Wald test for the statistical difference between two or more activitiy level estimates.

Usage

```
compareAct(fits)
```

Arguments

fits	A list of fitted actmod objects

Details

Uses a Wald test to ask whether the difference between estimates a1 and a2 is significantly different from 0: statistic W = (a1-a2)^2 / (SE1^2+SE2^2) tested on chi-sq distribution with 1 degree of freedom.

Value

A matrix with 4 columns: 1. differences between estimates; 2. SEs of the differences; 3. Wald statistics; 4. p-values (H0 is no difference between estimates). Matrix rows give all possible pairwise comparisons, numbered in the order in which they entered in the list fits.

Examples

```
#Test whether paca have a sigificantly different activity level from rat.
#Bootstrap reps limited to speed up example.
data(BCItime)
tPaca <- 2*pi*BCItime$time[BCItime$species=="ocelot"]
tRat <- 2*pi*BCItime$time[BCItime$species=="rat"]
fPaca <- fitact(tPaca, sample="data", reps=10)
fRat <- fitact(tRat, sample="data", reps=10)
fPaca@act
fRat@act
compareAct(list(fPaca,fRat))
```

---

compareCkern *Compare circular distributions.*

Description

Randomisation test for the probability that two sets of circular observations come from the same distribution.

Usage

```
compareCkern(fit1, fit2, reps = 999)
```

Arguments

fit1, fit2	Fitted activity models of class actmod created using function fitact.
reps	Number of bootstrap iterations.

Details

Calculates overlap index Dhat4 (see references) for the two fitted distributions, then generates a null distribution of overlap indices using data sampled randomly with replacement from the combined data. This randomised distribution is then used to define an empirical probability distribution against which the probability that the observed overlap arose by chance is judged. When one or both fitted models use weighted distributions, sampling probabilities are taken from the weights. If both models are weighted, the weights must therefore be on the same scale.

Value

A named 4-element vector: obs = observed overlap index; null = mean null overlap index; seNull = standard error of the null distribution; pNull = probability observed index arose by chance.

References

Ridout, M.S. & Linkie, M. (2009) Estimating overlap of daily activity patterns from camera trap data. Journal of Agricultural Biological and Environmental Statistics, 14, 322-337.

Examples

```
#Example with bootstrap reps limited to reduce run time
data(BCItime)
tPaca <- 2*pi*BCItime$time[BCItime$species=="paca"]
tRat <- 2*pi*BCItime$time[BCItime$species=="rat"]
fPaca <- fitact(tPaca)
fRat <- fitact(tRat)
compareCkern(fPaca,fRat,reps=10)
```

---

compareTimes *Compare activity between times of day*

Description

Uses a Wald test to statistically compare activity levels at given radian times of day for a fitted activity distribution.

Usage

```
compareTimes(fit, times)
```

Arguments

`fit`	Fitted `actmod` object with errors boostrapped (fit using `fitact` with `sample` argument != "none").
`times`	Numeric vector of radian times of day at which to compare activity levels. All pairwise comparisons are made.

Details

Bootrapping the activity model yields standard error estimates for the PDF. This function uses these SEs to compute a Wald statistic for the difference between PDF values (by inference activity levels) at given times of day: statistic W = (a1-a2)^2 / (SE1^2+SE2^2) tested on chi-sq distribution with 1 degree of freedom.

Value

A matrix with 4 columns: 1. differences between PDF values; 2. SEs of the differences; 3. Wald statistics; 4. p-values (H0 is no difference between estimates). Matrix rows give all possible pairwise comparisons, numbered in the order in which they appear in vector `times`.

Examples

```
data(BCItime)
tPaca <- 2*pi*BCItime$time[BCItime$species=="paca"]
fPaca <- fitact(tPaca, sample="data", reps=10)
plot(fPaca)
compareTimes(fPaca, c(5.5,6,0.5,1))
```

`density2` *Modified kernel density function*

Description

Modifies `stats::density` by: Adding SE and 95% confidence intervals for the density to the output; and Truncating calculation (not just reporting) of density values on from and/or to.

Usage

```
density2(x, reps = 999, ...)
```

Arguments

`x`	numeric data vector
`reps`	bootstrap iterations for SE/interval calculation; set to NULL to suppress
`...`	Additional arguments passed to `stas::density`

Details

Truncation copes with cases where no data are available outside truncation points. Truncation is achieved by fitting the density to the data augmented by reflecting it across each bound using the optimal bandwidth for the unaugmented data, and returning the resulting densities for the region between the bounds.

Value

A list with the same components as `stats::density` output plus: `se`: standard error of the density `lcl`, `ucl`: lower and upper 95% confidence intervals of the density

Examples

```
data(BCItime)
tm <- subset(BCItime, species=="ocelot")$time
dens <- density2(tm, from=0.25, to=0.75)
plot(dens$x, dens$y, type="l")
```

`dvmkern` *Circular kernel probability density function.*

Description

Optionally weighted Von Mises kernel probability densities.

Usage

```
dvmkern(x, dat, wt = NULL, bw = NULL, adj = 1)
```

Arguments

`x`	Numeric vector of radian times at which to evaluate the PDF.
`dat`	Numeric vector of radian time data to which the PDF is fitted.
`wt`	A numeric vector of weights for each `dat` value.
`bw`	Numeric value for kernel bandwidth.
`adj`	Numeric kernel bandwidth multiplier.

Details

If `bw` not provided it is calculated internally using `bw.calc`. The `adj` argument is used to adjust `bw` to facilitate exploration of fit flexibility.

Value

Numeric vector of probability densities evaluated at `x`.

See Also

`bwcalc`

Examples

```
#Example with made up input
tt <- runif(100,0,2*pi)
xx <- seq(0,2*pi, pi/256)
pdf <- dvmkern(xx, tt)
plot(xx, pdf, type="l")
```

---

dvonm *von Mises density function*

Description

Probability density function for the von Mises circular distribution.

Usage

```
dvonm(x, mu, k, log = FALSE)
```

Arguments

x	numeric angles (assumed to be radian).
mu	numeric, the mean direction of the distribution.
k	non-negative numeric, the concentration parameter distribution (kappa).
log	if TRUE log probabilities are returned.

Details

If more than one of x, mu and k have length > 1, values are recycled.

Value

Probability density value(s).

Examples

```
dvonm(seq(0, 2*pi, len=10), pi, 1)
```

`fitact` *Fit activity model to time-of-day data*

Description

Fits kernel density to radian time-of-day data and estimates activity level from this distribution. Optionally: 1. bootstraps the distribution, in which case SEs and confidence limits are also stored for activity level and PDF; 2. weights the distribution; 3. truncates the distribution at given times.

Usage

```
fitact(
  dat,
  wt = NULL,
  reps = 999,
  bw = NULL,
  adj = 1,
  sample = c("none", "data", "model"),
  bounds = NULL,
  show = TRUE
)
```

Arguments

`dat`	A numeric vector of radian time-of-day data.
`wt`	A numeric vector of weights for each `dat` value.
`reps`	Number of bootstrap iterations to perform. Ignored if `sample=="none"`.
`bw`	Numeric value for kernel bandwidth. If NULL, calculated internally.
`adj`	Numeric bandwidth adjustment multiplier.
`sample`	Character string defining sampling method for bootstrapping errors (see details).
`bounds`	A two-element vector defining radian bounds at which to truncate.
`show`	Logical whether or not to show a progress bar while bootstrapping.

Details

When no `bounds` are given (default), a circular kernel distribution is fitted using `dvmkern`. Otherwise, a normal kernel distribution is used, truncated at the values of `bounds`, using `density2`.

The bandwidth adjustment multiplier `adj` is provided to allow exploration of the effect of adjusting the internally calculated bandwidth on accuracy of activity level estimates.

The alternative bootstrapping methods defined by `sample` are:

- `"none"`: no bootstrapping
- `"data"`: sample from the data
- `"model"`: sample from the fitted probability density distribution

It's generally better to sample from the data, but sampling from the fitted distribution can sometimes provide more sensible confidence intervals when the number of observations is very small.

Value

An object of type actmod

Examples

```
#Fit without confidence limits
data(BCItime)
tm <- 2*pi*subset(BCItime, species=="brocket")$time
mod1 <- fitact(tm)
plot(mod1)

#Fit with confidence limits (limited reps to speed up)
mod2 <- fitact(tm, sample="data", reps=10)
plot(mod2)

#Fit weighted function to correct for detection radius 1.2 times higher
#by day than by night, assuming day between pi/2 (6 am) and pi*2/3 (6 pm)
weight <- 1/ifelse(tm>pi/2 & tm<pi*3/2, 1.2, 1)
mod3 <- fitact(tm, wt=weight)
plot(mod3)
#Overplot unweighted version for comparison
plot(mod1, add=TRUE, tline=list(col=2))

#Fit truncated function to consider only night time records,
#assuming night between pi*3/2 (6 pm) and pi/3 (6 am)
mod4 <- fitact(tm, bounds=c(pi*3/2, pi/2))
plot(mod4, centre="night")
```

---

fitlincirc *Linear-circular regression*

Description

Fits a Von Mises kernel distribution describing a linear variable as a function of a circular predictor, and boostraps the null distribution in order to evaluate significance of radial variation in the linear variable.

Usage

```
fitlincirc(circdat, lindat, pCI = 0.95, reps = 10, res = 512)
```

Arguments

circdat	Numeric vector of radian data matched with lindat.
lindat	Numeric vector of linear data matched with circdat.
pCI	Single numeric value between 0 and 1 defining proportional confidence interval to return.
reps	Integer number of bootstrap repetitions to perform.

`res` Resolution of fitted distribution and null confidence interval - specifically a single integer number of points on the circular scale at which to record distributions.

Details

Deviation of `lindat` from the null expecation is assessed either visually by the degree to which the fitted distribution departs from the null confidence interval (use generic plot function), or quantitatively by column `p` of slot `fit` in the resulting `lincircmod-class` object.

Value

An object of type `lincircmod-class`

References

Xu, H., Nichols, K. & Schoenberg, F.P. (2011) Directional kernel regression for wind and fire data. Forest Science, 57, 343-352.

Examples

```
#Example with reps limited to increase speed
data(BCIspeed)
i <- BCIspeed$species=="ocelot"
sp <- log(BCIspeed$speed[i])
tm <- BCIspeed$time[i]*2*pi
mod <- fitlincirc(tm, sp, reps=50)
plot(mod, CircScale=24, xaxp=c(0,24,4), xlab="Time", ylab="log(speed)")
legend(8,-3, c("Fitted speed", "Null CI"), col=1:2, lty=1:2)
```

---

gettime *Convert time of day data to numeric*

Description

Accepts data of class POSIXct, POSIXlt or character and returns the time of day element as numeric (any date element is ignored).

Usage

```
gettime(
  x,
  scale = c("radian", "hour", "proportion"),
  ...,
  tryFormats = c("%Y-%m-%d %H:%M:%OS", "%Y/%m/%d %H:%M:%OS",
    "%Y:%m:%d %H:%M:%OS", "%Y-%m-%d %H:%M", "%Y/%m/%d %H:%M",
    "%Y:%m:%d %H:%M", "%Y-%m-%d", "%Y/%m/%d", "%Y:%m:%d")
)
```

Arguments

x	A vector of POSIXct, POSIXlt or character format time data to convert.
scale	The scale on which to return times (see Value for options).
...	arguments passed to as.POSIXlt
tryFormats	formats to try when converting date from character, passed to as.POSIXlt

Value

A vector of numeric times of day in units defined by the `scale` argument: radian, on the range [0, 2*pi]; hours, on the range [0, 24]; proportion, on the range [0, 1].

See Also

`strptime`

Examples

```
data(BCItime)
rtime <- gettime(BCItime$date)
htime <- gettime(BCItime$date, "hour")
ptime <- gettime(BCItime$date, "proportion")
summary(rtime)
summary(htime)
summary(ptime)
```

---

get_suntimes *Calculates solar event times*

Description

Calculates approximate times of sunrise and sunset and day lengths for given dates at given locations.

Usage

```
get_suntimes(
  date,
  lat,
  lon,
  offset,
  ...,
  tryFormats = c("%Y-%m-%d %H:%M:%OS", "%Y/%m/%d %H:%M:%OS",
    "%Y:%m:%d %H:%M:%OS", "%Y-%m-%d %H:%M", "%Y/%m/%d %H:%M",
    "%Y:%m:%d %H:%M", "%Y-%m-%d", "%Y/%m/%d", "%Y:%m:%d")
)
```

Arguments

date	character, POSIX or Date format date/time value(s)
lat, lon	latitude and longitude in decimal degrees
offset	the time offset in hours relative to UTC (GMT) for results
...	arguments passed to as.POSIXlt
tryFormats	formats to try when converting date from character, passed to as.POSIXlt

Details

Function adapted from https://www.r-bloggers.com/2014/09/seeing-the-daylight-with-r/

Value

A dataframe with columns sunrise and sunset (given in the timezone defined by offset) and daylength, all expressed in hours.

References

Teets, D.A. 2003. Predicting sunrise and sunset times. The College Mathematics Journal 34(4):317-321.

Examples

```
data(BCItime)
dat <- subset(BCItime, species=="ocelot")$date
get_suntimes(dat, 9.156335, -79.847682, -5)
```

---

lincircKern *Linear-circular kernel fit*

Description

Fits a Von Mises kernel distribution describing a linear variable as a function of a circular predictor.

Usage

```
lincircKern(x, circdat, lindat)
```

Arguments

x	Numeric vector of radian values at which to evaluate the distribution.
circdat	Numeric vector of radian data matched with lindat.
lindat	Numeric vector of linear data matched with circdat.

Value

A numeric vector of fitted lindat values matched with x.

References

Xu, H., Nichols, K. & Schoenberg, F.P. (2011) Directional kernel regression for wind and fire data. Forest Science, 57, 343-352.

Examples

```
data(BCIspeed)
i <- BCIspeed$species=="ocelot"
log_speed <- log(BCIspeed$speed[i])
time <- BCIspeed$time[i]*2*pi
circseq <- seq(0,2*pi,pi/256)
trend <- lincircKern(circseq, time, log_speed)
plot(time, log_speed, xlim=c(0, 2*pi))
lines(circseq, trend)
```

---

lincircmod-class *An S4 class describing linear-circular relationships.*

Description

An S4 class describing linear-circular relationships.

Slots

`data` Object of class `"data.frame"`, the input data, with columns `lindat` (linear data) and `circdat` (circular data).

`fit` Object of class `"data.frame"`, summary of the model fit, with columns: `x`: A regular ascending sequence from 0 to 2*pi at which other columns evaluated; `fit`: The linear fitted values; `p`: The two tailed probability of observing the fitted values under a random (null) circular distribution; `nullLCL`: The lower 95% confidence limit of the null distribution; `nullUCL`: The upper 95% confidence limit of the null distribution.

---

ovl4 *Index of overlap between circular distributions.*

Description

Calculates Dhat4 overlap index (see reference) between two kernel distributions.

Usage

```
ovl4(fit1, fit2)
```

Arguments

`fit1`, `fit2`	Fitted activity models of class actmod created using function fitact.

Details

Uses linear interpolation to impute values from kernel distributions.

Value

Scalar overlap index (specifically Dhat4).

References

Ridout, M.S. & Linkie, M. (2009) Estimating overlap of daily activity patterns from camera trap data. Journal of Agricultural Biological and Environmental Statistics, 14, 322-337.

Examples

```
data(BCItime)
oceAct <- fitact(subset(BCItime, species=="ocelot")$time*2*pi)
broAct <- fitact(subset(BCItime, species=="brocket")$time*2*pi)
ovl4(oceAct, broAct)
```

`plot.actmod` *Plot activity distribution*

Description

Plot an activity probability distribution from a fitted `actmod` object.

Usage

```
## S3 method for class 'actmod'
plot(
  x,
  xunit = c("clock", "hours", "radians"),
  yunit = c("frequency", "density"),
  data = c("histogram", "rug", "both", "none"),
  centre = c("day", "night"),
  dline = list(lwd = ifelse(data == "rug", 0.1, 1)),
  tline = NULL,
  cline = list(lty = 2),
  add = FALSE,
  xaxis = NULL,
  ...
)
```

Arguments

x	Object of class actmod.
xunit	Character string defining x-axis unit.
yunit	Character string defining y-axis unit.
data	Character string defining whether to plot the data distribution and if so which style to use.
centre	Character string defining whether to centre the plot on midday or midnight.
dline	List of plotting parameters for data lines.
tline	List of plotting parameters for trend line.
cline	List of plotting parameters for trend confidence interval lines.
add	Logical defining whether to create a new plot (default) or add to an existing plot.
xaxis	List of plotting parameters to pass to axis command for x-axis plot (see axis for arguments).
...	Additional arguments passed to internal plot call affecting only the plot frame and y axis. Modify x axis through xaxis.

Details

When xunit=="clock", The underlying numeric range of the x-axis is [0,24] if centre=="day", or [-12,12] if centre=="night".

Value

No return value, called to create a plot visualising an activity model.

Examples

```
data(BCItime)
otm <- 2*pi*subset(BCItime, species=="ocelot")$time
btm <- 2*pi*subset(BCItime, species=="brocket")$time
omod <- fitact(otm)
bmod <- fitact(btm)
plot(omod, yunit="density", data="none")
plot(bmod, yunit="density", data="none", add=TRUE, tline=list(col="red"))
legend("topleft", c("Ocelot", "Brocket deer"), col=1:2, lty=1)

mod <- fitact(otm, sample="data", reps=10)
plot(mod, dline=list(col="grey"),
          tline=list(col="red", lwd=2),
          cline=list(col="red", lty=3))

mod2 <- fitact(otm, bounds=c(pi*3/2, pi/2))
plot(mod2, centre="night")
plot(mod2, centre="night", xlim=c(-6,6), xaxis=list(at=seq(-6,6,2)))
```

`plot.lincircmod` *Plot linear-circular relationship*

Description

Plot linear against circular data along with the fitted and null confidence limit distributions from a fitted `lincircmod` object.

Usage

```
## S3 method for class 'lincircmod'
plot(
  x,
  CircScale = 2 * pi,
  tlim = c(0, 1),
  fcol = "black",
  flty = 1,
  ncol = "red",
  nlty = 2,
  ...
)
```

Arguments

`x`	Object of class `lincircmod`.
`CircScale`	Single numeric value defining the plotting maximum of the circular scale.
`tlim`	Numeric vector with two elements >=0 and <=1 defining the lower and upper limits at which to plot distributions; default plots the full range.
`fcol`, `flty`, `ncol`, `nlty`	Define line colour (`col`) and type (`lty`) for fitted (`f`) and null (`n`) distributions; input types as for `col` and `lty`, see `par`.
`...`	Additional arguments passed to the inital plot construction, affecting axes and data plot symbols.

Value

No return value, called to create a plot visualising a linear-circular relationship.

redf *Random numbers from empirical distribution function.*

Description

Random numbers drawn from an empirical distribution defined by paired values and probabilities.

Usage

```
redf(n, fit)
```

Arguments

n	Integer number of random numbers to return.
fit	Data frame defining the emprical distribution (see details).

Details

The distribution function is defined by `fit`, which must be a dataframe containing (at least) columns named: x: a regular sequence of values from which to draw; y: corresponding pdf values.

Value

A numeric vector.

Examples

```
data(BCItime)
tm <- 2*pi*subset(BCItime, species=="paca")$time
mod <- fitact(tm)
rn <- redf(1000, as.data.frame(mod@pdf))
```

solartime *Transforms clock time to solar time anchored to sun rise and sunset times for a given location.*

Description

This is a wrapper for `transtime` that takes non-numeric date-time input together with latitude and longitude to calculate mean average sunrise and sunset times, which are then used to anchor the transformation using average anchoring.

Usage

```
solartime(
  dat,
  lat,
  lon,
  tz,
  ...,
  tryFormats = c("%Y-%m-%d %H:%M:%OS", "%Y/%m/%d %H:%M:%OS",
    "%Y:%m:%d %H:%M:%OS", "%Y-%m-%d %H:%M", "%Y/%m/%d %H:%M",
    "%Y:%m:%d %H:%M", "%Y-%m-%d", "%Y/%m/%d", "%Y:%m:%d")
)
```

Arguments

`dat`	A vector of character, POSIXct or POSIXlt date-time values.
`lat`, `lon`	Single numeric values or numeric vectors the same length as `dat` giving site latitude and longitude in decimal format.
`tz`	A single numeric value or numeric vector same length as `dat` giving time zone (see Details).
`...`	arguments passed to as.POSIXlt
`tryFormats`	formats to try when converting date from character, passed to as.POSIXlt

Details

Time zone `tz` should be expressed in numeric hours relative to UTC (GMT).

Value

A list with elements:

`input`: event input dates-times in POSIXlt format.

`clock`: radian clock time data.

`solar`: radian solar time data anchored to average sun rise and sun set times.

References

Vazquez, C., Rowcliffe, J.M., Spoelstra, K. and Jansen, P.A. in press. Comparing diel activity patterns of wildlife across latitudes and seasons: time transformation using day length. Methods in Ecology and Evolution.

See Also

`strptime`, `transtime`

Examples

```
data(BCItime)
subdat <- subset(BCItime, species=="ocelot")
times <- solartime(subdat$date, 9.156335, -79.847682, -5)
rawAct <- fitact(times$clock)
avgAct <- fitact(times$solar)
plot(rawAct)
plot(avgAct, add=TRUE, data="n", tline=list(col="cyan"))
```

---

transtime *Transforms clock time to solar times.*

Description

Transforms time expressed relative to either the time of a single solar event (anchor times - Nouvellet et al. 2012), or two solar events (such as sun rise and sun set - Vazquez et al. in press).

Usage

```
transtime(
  dat,
  anchor,
  mnanchor = NULL,
  type = c("average", "equinoctial", "single")
)
```

Arguments

dat	A vector of radian event clock times.
anchor	A vector or matrix matched with dat containing radian anchor times on the day of each event (see Details).
mnanchor	A scalar or two-element vector of numeric radian mean anchor times (see Details).
type	The type of transformation to use (see Details).

Details

If double anchoring is requested (i.e. type is equinoctial or average), the anchor argument requires a two-column matrix, otherwise a vector. The argument mnanchor can usually be left at its default NULL value. In this case, the mean anchors are set to c(pi/2, pi*3/2) when type=="equinoctial", otherwise the anchor mean(s).

Although the anchors for transformation are usually likely to be solar events (e.g. sun rise and/or sunset), they could be other celestial (e.g. lunar) or human-related (e.g. timing of artificial lighting) events.

Value

A vector of radian transformed times.

References

Vazquez, C., Rowcliffe, J.M., Spoelstra, K. and Jansen, P.A. in press. Comparing diel activity patterns of wildlife across latitudes and seasons: time transformation using day length. Methods in Ecology and Evolution.

Nouvellet, P., Rasmussen, G.S.A., Macdonald, D.W. and Courchamp, F. 2012. Noisy clocks and silent sunrises: measurement methods of daily activity pattern. Journal of Zoology 286: 179-184.

Examples

```
data(BCItime)
subdat <- subset(BCItime, species=="ocelot")
suntimes <- pi/12 * get_suntimes(subdat$date, 9.156335, -79.847682, -5)[, -3]
rawtimes <- subdat$time*2*pi
avgtimes <- transtime(rawtimes, suntimes)
eqntimes <- transtime(rawtimes, suntimes, type="equinoctial")
sngtimes <- transtime(rawtimes, suntimes[,1], type="single")
rawAct <- fitact(rawtimes)
avgAct <- fitact(avgtimes)
eqnAct <- fitact(eqntimes)
sngAct <- fitact(sngtimes)
plot(rawAct)
plot(avgAct, add=TRUE, data="n", tline=list(col="magenta"))
plot(eqnAct, add=TRUE, data="n", tline=list(col="orange"))
plot(sngAct, add=TRUE, data="n", tline=list(col="cyan"))
```

`trigmolen` *title trigonometric moment length*

Description

Calculate trigonometric moment length

Usage

```
trigmolen(x, p)
```

Arguments

`x`	a vector of circular values, assumed to be radian.
`p`	order of trigonometric moment to be computed.

Value

Trigonometric moment length of the input.

wrap *Wraps data on a given range.*

Description

Input data outside the given bounds (default radian [0, 2*pi]) are wrapped to appear within the range.

Usage

```
wrap(x, bounds = c(0, 2 * pi))
```

Arguments

x	A vector of numeric data.
bounds	The range within which to wrap x values

Details

As an example of wrapping, on bounds [0, 1], a value of 1.2 will be converted to 0.2, while a value of -0.2 will be converted to 0.8.

Value

A vector of numeric values within the limits defined by bounds

Examples

```
data(BCItime)
adjtime <- BCItime$time + 1/24
summary(adjtime)
adjtime <- wrap(adjtime, c(0,1))
summary(adjtime)
```

Index