Package 'skyscapeR'

October 21, 2017

Type Package

Title Skyscape Archaeology Data Reduction, Visualization and Analysis

Version 0.2.2

Date 2017-10-21

Author Fabio Silva

Maintainer Fabio Silva <f.silva@uwtsd.ac.uk>

Description A toolset for data reduction, visualization and analysis in skyscape archaeology, archaeoastronomy and cultural astronomy.

Depends R (>= 2.10), astrolibR

Imports pracma, png, plotrix, oce, utils, stats, graphics, grDevices, MESS, RColorBrewer, numDeriv, parallel, foreach, doParallel, rootSolve

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 6.0.1

Suggests testthat, knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2017-10-21 14:52:44 UTC

R topics documented:

`antizenith` *Declination of the anti-zenith sun for a given location*

Description

This function returns the declination of the sun when it is at the anti-zenith, or nadir, for a given location. If this phenomena does not occur at given location (i.e. if location is outside the tropical band) the function returns a *NULL* value.

Usage

```
antizenith(loc)
```

Arguments

loc	This can be either the latitude of the location, or a *skyscapeR.horizon* object.

See Also

jS, dS, eq, zenith

Examples

```
# Anti-zenith sun declination for Mexico City:
antizenith(19.419)

# There is no anti-zenith sun phenomena in London:
antizenith(51.507)
```

---

az2dec *Calculates declination from azimuth and altitude measurements*

Description

This function calculates the declination corresponding to an orientation , i.e. an azimuth. The altitude can either be given or, alternatively, if a *skyscapeR.horizon* object is provided, the corresponding horizon altitude will be automatically retrieved. This function is a wrapper for function hor2eq of package *astrolibR*.

Usage

```
az2dec(az, loc, alt, ...)
```

Arguments

az	Azimuth(s) for which to calculate declination(s). See examples below.
loc	Location, can be either a *skyscapeR.horizon* object or, alternatively, a latitude.
alt	Altitude of orientation. Optional, if left empty and a skyscapeR.object is provided then this is will automatically retrieved from the horizon data via hor2alt
...	Any other parameters to be passed unto hor2eq.

See Also

hor2eq, hor2alt

Examples

```
hor <- download.HWT('HIFVTBGK')

dec <- az2dec(92, hor)
dec <- az2dec(92, hor, alt=4)

# Can also be used for an array of azimuths:
decs <- az2dec( c(87,92,110), hor )
```

---

createHor *Create* .skyscapeR.horizon *object from Az/Alt data*

Description

This function creates a *skyscapeR.horizon* object from measurements of azimuth and altitude.

Usage

```
createHor(az, alt, loc, name)
```

Arguments

az	Array of azimuth values
alt	Array of altitude values.
loc	Location, a vector containing the latitude and longitude of the location, in this order.
name	Name of site.

See Also

plotHor

Examples

```
# Create a skyscapeR.horizon from 5 measurements:
az <- c(0,90,180,270,360)
alt <- c(0,5,5,0,0)
hor <- createHor(az, alt, c(40.1,-8), 'Test')
plotHor(hor)
```

curvigram *Computes declination curvigram*

Description

This function computes the curvigram of declinations, using provided measurement uncertainty and a Gaussian kernel, i.e. using the method of Silva (2017). When all measurements have the same associated uncertainty this function wraps density and accepts the same input for *bw* in *unc*.

Usage

```
curvigram(dec, unc = 2, norm = F, cut = 4, range, n = 512)
```

Arguments

dec	Array of declination values
unc	(Optional) Either a single value or string to be applied to all measurements (see bw.nrd), or an array of values of the same length as *dec*. Defaults to 2 degrees.
norm	(Optional) Boolean specifying whether the resulting curvigram should be normalized to unity. Defaults to *FALSE*.
cut	(Optional) Number of uncertainties beyond the extremes of the data at which to trim the curvigram. Defaults to 4. See density.
range	(Optional) As an alternative to *cut* you can stipulate the range of declination values to output as an array of two values. See *from, to* in density.
n	(Optional) The number of equally spaced points at which the curvigram is to be calculated. Defaults to 512. See *n* in density.

References

Silva, Fabio (2017) Inferring Alignments I: Exploring the Accuracy and Precision of Two Statistical Approaches, *Journal of Skyscape Archaeology* 3(1), 93-111. DOI: 10.1558/jsa.31958

See Also

density

Examples

```
# Curvigram of Ruggles' Recumbent Stone Circle data:
data(RugglesRSC)
curv <- curvigram(RugglesRSC$Dec, 2)
plotCurv(curv)
```

download.HWT *Download horizon data from* HeyWhatsThat

Description

This function downloads horizon data from *HeyWhatsThat*, given its ID, and saves it as a *skyscapeR.horizon* object.

Usage

```
download.HWT(HWTID)
```

Arguments

HWTID	This is the 8 character ID attributed by *HeyWhatsThat.com*

References

HeyWhatsThat.com

Examples

```
# Retrieve horizon data for \href{https://www.heywhatsthat.com/?view=HIFVTBGK}{Liverpool Cathedral}:
hor <- download.HWT('HIFVTBGK')
```

dS *Declination of December Solstice for a given year*

Description

This function calculates the declination of the sun at December Solstice for a given year, based upon obliquity estimation.

Usage

```
dS(year = cur.year)
```

Arguments

year	Year for which to calculate the declination. Defaults to present year as given by *Sys.Date()*.

See Also

obliquity, jS, eq, zenith, antizenith

Examples

```
# December Solstice declination for year 3999 BC:
dS(-4000)
```

---

eq *Declination of sun at the equinoxes*

Description

This function always returns a value of zero, which is the declination of the sun on the day of the (astronomical) equinoxes.

Usage

```
eq(bh = NULL)
```

Arguments

bh	*NULL* parameter. Can be left empty.

See Also

jS, dS, zenith, antizenith

Examples

```
eq()
```

---

exportHor *Exports a* skyscapeR.horizon *object into* Stellarium *format*

Description

This function exports any *skyscapeR.horizon* object into the landscape format of *Stellarium*, ready to be imported.

Usage

```
exportHor(hor, name, author = "skyscapeR", description, ground_col, hor_col)
```

Arguments

hor	Horizon data in *skyscapeR.horizon* format.
name	Horizon name to be displayed in *Stellarium*, if different from one in *skyscapeR.horizon* object.
author	(Optional) Author, to be included in *landscape.ini* file.
description	(Optional) Description, to be included in *landscape.ini* file.
ground_col	Colour of ground. Defaults to *Stellarium*'s default.
hor_col	Colour of horizon line. Defaults to *Stellarium*'s default.

References

Stellarium: a free open source planetarium

See Also

createHor, download.HWT, plotHor

Examples

```
# Downloads horizon data from HeyWhatsThat and exports it into Stellarium:
hor <- download.HWT('HIFVTBGK')
exportHor(hor, name='Test', description='Test horizon export to Stellarium')
```

---

hor2alt *Retrieves horizon altitude for a given azimuth from a given horizon profile*

Description

This function retrieves the horizon altitude for a given azimuth from a previously created *skyscapeR.horizon* object via spline interpolation.

Usage

```
hor2alt(hor, az)
```

Arguments

hor	A *skyscapeR.horizon* object from which to retrieve horizon altitude.
az	Array of azimuth(s) for which to retrieve horizon altitude(s).

See Also

createHor, download.HWT

Examples

```
hor <- download.HWT('HIFVTBGK')
hor2alt(hor, 90)
```

---

jS *Declination of June Solstice for a given year*

Description

This function calculates the declination of the sun at June Solstice for a given year, based upon obliquity estimation.

Usage

```
jS(year = cur.year)
```

Arguments

year	Year for which to calculate the declination. Defaults to present year as given by *Sys.Date()*.

See Also

obliquity, dS, eq, zenith, antizenith

Examples

```
# June Solstice declination for year 3999 BC:
jS(-4000)
```

Laskar04 *Astronomical elements from Laskar et al. (2004)*

Description

Astronomical elements (longitude of perihelion, obliquity and eccentricity) by step of 1ka, from -51M BP to present (la04past) and from present to + 21M BP (la04future).

Usage

```
data(Laskar04)
```

Format

A list of two data frames with 4 variables:

time Time in years before or after J1950.0

ecc Eccentricity

eps Obliquity

varpi Longitude of Perihelion

References

Laskar, J. et al. (2004), A long-term numerical solution for the insolation quantities of the Earth, *Astron. Astroph.*, 428, 261-285, doi:10.1051/0004-6361:20041335.

Michel Crucifix (2016). palinsol: Insolation for Palaeoclimate Studies. R package version 0.93. [CRAN page](https://CRAN.R-project.org/package=palinsol)

Examples

```
data(Laskar04)
```

`mag.dec` *Estimates magnetics declination (diff. between true and magnetic north) based on IGRF 12th gen model*

Description

This function estimates the magnetic declination at a given location and moment in time, using the *12th generation International Geomagnetic Reference Field (IGRF)* model. This function is a wrapper for function `magneticField` of package *oce*.

Usage

```
mag.dec(loc, date)
```

Arguments

`loc`	Location, can be either a *skyscapeR.horizon* object or, alternatively, a latitude.
`date`	Date for which to calculate magnetic declination in the format: 'YYYY/MM/DD'

See Also

`magneticField`

Examples

```
# Magnetic Declination for London on April 1st 2016:
london.lat <- 51.5074 #N
london.lon <- -0.1278 #W
loc <- c( london.lat, london.lon )
mag.dec( loc, "2016/04/01" )
```

nh.LunarRange — *Declination distribution of the Moon throughout the year for null hypothesis significance testing*

Description

This function returns the declination distribution of the the Sun throughout the year for use in significance testing.

Usage

```
nh.LunarRange(year = cur.year)
```

Arguments

year	Year for which to calculate the distribution. Defaults to present year as given by Sys.Date()

See Also

sigTest, nh.Uniform, nh.SummerFM

Examples

```
## Not run:
aux <- nh.SolarRange(-4000)
plot(aux$dec, aux$density, type='l')

## End(Not run)
```

nh.SolarRange — *Declination distribution of the Sun throughout the year for null hypothesis significance testing*

Description

This function returns the declination distribution of the the Sun throughout the year for use in significance testing.

Usage

```
nh.SolarRange(year = cur.year)
```

Arguments

year	Year for which to calculate the distribution. Defaults to present year as given by Sys.Date()

See Also

sigTest, nh.Uniform, nh.SummerFM

Examples

```
## Not run:
aux <- nh.SolarRange(-4000)
plot(aux$dec, aux$density, type='l')

## End(Not run)
```

---

nh.SummerFM *Declination distribution of the Summer Full Moon for null hypothesis significance testing*

Description

This function returns the declination distribution of the Summer Full Moon for use in significance testing.

Usage

```
nh.SummerFM(min.phase = 0.99, min.sundec = 20, year = cur.year)
```

Arguments

min.phase	(Optional) This should be the minimum lunar phase (i.e. percentage illumination) for the moon to be considered full. The value should range between 0 (dark moon) and 1 (full moon). Defaults to 0.99.
min.sundec	(Optional) This should be the minimum solar declination for the moon to be considered a *summmer* full moon. Defaults to 20 degrees, corresponding to a month before or after june solstice.
year	Year for which to calculate the obliquity. Defaults to present year as given by Sys.Date()

See Also

sigTest, nh.Uniform, nh.SolarRange

Examples

```
## Not run:
aux <- nh.SummerFM(.99, 20, -4000)
plot(aux$dec, aux$density, type='l')

## End(Not run)
```

nh.Uniform — *Declination distribution corresponding to a uniform azimuthal distribution for null hypothesis significance testing*

Description

This function returns the declination distribution that corresponds to a uniform distribution in azimuths (i.e. random orientation) for use in significance testing.

Usage

```
nh.Uniform(loc, alt = 0)
```

Arguments

loc	This can be either the latitude of the location, or a *skyscapeR.horizon* object.
alt	(Optional) The horizon altitude to use in az2dec conversion. Defaults to 0 degrees.

See Also

sigTest, nh.SummerFM, nh.SolarRange

Examples

```
## Not run:
aux <- nh.Uniform(loc=c(52,-2), alt=2)
plot(aux$dec, aux$density, type='l')

## End(Not run)
```

nMjLX — *Declination of northern major Lunar Extreme for a given year*

Description

This function calculates the declination of the northern major Lunar Extreme for a given year, by simple addition of obliquity with maximum lunar inclination.

Usage

```
nMjLX(year = cur.year)
```

Arguments

year	Year for which to calculate the declination. Defaults to present year as given by *Sys.Date()*.

See Also

nmnLX, smnLX, sMjLX

Examples

```
# Northern major Lunar Extreme declination for year 2499 BC:
nMjLX(-2500)
```

---

nmnLX *Declination of northern minor Lunar Extreme for a given year*

Description

This function calculates the declination of the northern minor Lunar Extreme for a given year, by simple addition of obliquity with maximum lunar inclination.

Usage

```
nmnLX(year = cur.year)
```

Arguments

year	Year for which to calculate the declination. Defaults to present year as given by *Sys.Date()*.

See Also

smnLX, nMjLX, sMjLX

Examples

```
# Northern minor Lunar Extreme declination for year 2499 BC:
nmnLX(-2500)
```

---

obliquity *Computes obliquity based on* Laskar et al (2004) *tables*

Description

This function calculates the obliquity for a given year, by interpolating the tables provided by *Laskar et al. (2004)*. It is a modified version of function la04 of package *palinsol*.

Usage

```
obliquity(year = cur.year)
```

Arguments

year	Year for which to calculate the obliquity. Defaults to present year as given by Sys.Date()

References

Laskar, J. et al. (2004), A long-term numerical solution for the insolation quantities of the Earth, *Astron. Astroph.*, 428, 261-285, doi:10.1051/0004-6361:20041335.

Laskar, J. et al. (2004), A long-term numerical solution for the insolation quantities of the Earth, *Astron. Astroph.*, 428, 261-285, doi:10.1051/0004-6361:20041335.

Michel Crucifix (2016). palinsol: Insolation for Palaeoclimate Studies. R package version 0.93. [CRAN page](https://CRAN.R-project.org/package=palinsol)

See Also

la04

Examples

```
#' # Obliquity for year 3999 BC:
obliquity(-4000)
```

---

orbit *Calculate visible path of celestial object at given location*

Description

This function calculates the visible path of a celestial object from any location on earth. It outputs a *skyscapeR.orbit* object, which includes AZ and ALT information.

Usage

```
orbit(dec, loc, res = 0.5, ...)
```

Arguments

dec	Declination of object.
loc	Location, either a *skyscapeR.object* or a vector containing the latitude and longitude of location, in this order.
res	The resolution (in degrees of RA) with which to calculate the path.
...	Any other parameters to be passed unto eq2hor.

Examples

```
# Visible path of sun on June Solstice on year 3999 BC from London:
sun.dec <- jS(-4000)
london.lat <- 51.5074 #N
london.lon <- -0.1278 #W
loc <- c( london.lat, london.lon )
path <- orbit(sun.dec, loc)
plot(path$az, path$alt, ylim=c(0,90), type='l', xlab='AZ', ylab='ALT', col='red', lwd=2)
```

---

plotAz *Polar plot of orientations (azimuths)*

Description

This function creates a polar plot of azimuthal data. It is a wrapper for `polar.plot`

Usage

```
plotAz(az, obj, loc, obj.label = T, ...)
```

Arguments

`az`	Array of azimuths or data frame with column named *True.Azimuth*. Values outside the [0, 360] range will be ignored.
`obj`	(Optional) A *skyscapeR.object* object created with `sky.objects` for displaying the azimuths of celestial objects. Beware that this assumes a single location (given by parameter loc) and a flat horizon of zero degrees.
`loc`	(Optional) This can be either the latitude of the location, or a *skyscapeR.horizon* object. Only necessary for plotting potential celestial targets.
`obj.label`	(Optional) Boolean to control whether to label the celestial objects in the polar plot. Defaults to *TRUE*.
`...`	Any other parameters to be passed unto `polar.plot`

See Also

`polar.plot`, `sky.objects`

Examples

```
# Plot some azimuth data:
az <- c(120, 100, 93, 97, 88, 115, 112, 67)
plotAz(az)

# To visualize this data against the common solar and lunar targets:
tt <- sky.objects(c('sun','moon'), epoch=-2000, lty=c(2,3))
plotAz(az, tt, loc=c(35,-8))
```

plotCurv *Plot a curvigram*

Description

This function creates a plot of a curvigram.

Usage

```
plotCurv(curv, obj, obj.label = T, signif, xlim = NULL, ...)
```

Arguments

curv	Object of *skyscapeR.curv* format, created using curvigram.
obj	(Optional) A *skyscapeR.object* object created with sky.objects for displaying the declination of celestial objects.
obj.label	(Optional) Boolean to control whether to label the celestial objects in the curvigram. Defaults to *TRUE*.
signif	(Optional) A *skyscapeR.sig* object created with sigTest for displaying confidence envelope around the chosen null hypothesis and overall p-value.
xlim	Array of two values restricting the horizontal range of the plot.
...	Any other parameters to be passed unto plot.default.

See Also

curvigram, sky.objects, sigTest

Examples

```
# Plot the curvigram of Recumbent Stone Circles:
data(RugglesRSC)
curv <- curvigram(RugglesRSC$Dec, unc=2)
plotCurv(curv, xlim=c(-40,0))

# Redo the plot to include lunar extreme declinations:
LEx <- sky.objects('moon', -2000, col='red', lty=2)
plotCurv(curv, objects=LEx, xlim=c(-40,0))

# Add significance testing information:
## Not run:
sig <- sigTest(curv, nh.Uniform(c(57,2)))
plotCurv(curv, objects=LEx, signif=sig, xlim=c(-40,0))

## End(Not run)
```

plotHor *Plot horizon data*

Description

This function creates a plot of horizon data.

Usage

```
plotHor(hor, show.az = F, max.alt, az0 = 0, zoom = F, obj, measure, ...)
```

Arguments

hor	Object of *skyscapeR.horizon* format.
show.az	Boolean that controls whether to display azimuth values on horizontal axis. Defaults to *FALSE*.
max.alt	Maximum altitude to display. Defaults to 45 degrees.
az0	Leftmost azimuth of plot. Defaults to 0 degrees, i.e. North at the left.
zoom	Boolean that controls whether to provide a zoomed-in view of 100 degrees in azimuth and 5 degrees of altitude above the horizon line. Defaults to *FALSE*.
obj	(Optional) A *skyscapeR.object* object created with sky.objects for displaying the paths of celestial objects.
measure	(Optional) A data.frame object with columns *True.Azimuth* and *Altitude* such as the ones produced by reduct.compass or reduct.theodolite. If no column named *Altitude* is found then it will plot all azimuths are zero degrees altitude.
...	Any other parameters to be passed unto plot.default.

See Also

download.HWT, sky.objects

Examples

```
# Plot a horizon retrieved from HeyWhatsThat:
hor <- download.HWT('HIFVTBGK')
plotHor(hor)

# Add the paths of the solstices and equinoxes sun in the year 1999 BC:
tt <- sky.objects('sun', -2000, 'blue')
plotHor(hor, objects=tt)
```

plotPhases *Plot stellar phase and seasonality*

Description

This function creates a plot of stellar seasonality and phases/events.

Usage

```
plotPhases(starphase, ...)
```

Arguments

starphase	Object of *skyscapeR.starphase* format.
...	Any other parameters to be passed unto plot.default.

See Also

star.phases

Examples

```
# Plot the seasonality of Aldebaran for 3999 BCE:
## Not run:
ss <- star.phases('Aldebaran',-4000, c(35,-8))
plotPhases(ss)

## End(Not run)
```

plotZscore *Plot a z-score transformed curvigram*

Description

This function creates a plot of a z-score transformed curvigram, which is to say the curvigram transformed into sigma units, based on a previously generated significance test.

Usage

```
plotZscore(signif, obj, obj.label = T, xlim = NULL)
```

Arguments

signif	A *skyscapeR.sig* object created with sigTest.
obj	(Optional) A *skyscapeR.object* object created with sky.objects for displaying the declination of celestial objects.
obj.label	(Optional) Boolean to control whether to label the celestial objects in the curvigram. Defaults to *TRUE*.
xlim	Array of two values restricting the horizontal range of the plot.

See Also

sigTest

Examples

```
## Not run:
data(RugglesRSC)
curv <- curvigram(RugglesRSC$Dec, unc=2)
sig <- sigTest(curv, nh.Uniform(c(57,2)))

plotZscore(sig)

## End(Not run)
```

---

reduct.compass *Data reduction for compass measurements*

Description

This function calculates the true azimuth of a structure measured with a compass.

Usage

```
reduct.compass(loc, mag.az, date, magdec, alt, name, ID, HWT.ID)
```

Arguments

loc	Location, either a *skyscapeR.object* or a vector containing the latitude and longitude of location, in this order.
mag.az	Array of magnetic azimuth measurements.
date	(Optional) Date of measurements as a string in the format: 'YYYY/MM/DD'. Only necessary is *magdec* is not given.
magdec	(Optional) Magnetic declination, if known.
alt	(Optional) Altitude, necessary for automatic declination calculation. If missing and *loc* is a *skyscapeR.horizon* object then the altitude will be automatically read from the horizon profile.

name	(Optional) Names or labels to identify each measurement.
ID	(Optional) IDs or codes to identify each measurement.
HWT.ID	(Optional) HeyWhatsThat IDs relating to a previously generated horizon profile for measurement.

See Also

mag.dec, az2dec, hor2alt

Examples

```
loc <- c(35,-7)
mag.az <- c(89.5, 105, 109.5)
data <- reduct.compass(loc, mag.az, "2016/04/02")

# Declination will be automatically calculated if the altitude is also given:
data <- reduct.compass(loc, mag.az, "2016/04/02", alt=c(1,2,0))

# Alternatively, the altitude can be automatically retrieved from a horizon profile:
hor <- download.HWT('NML6GMSX')
data <- reduct.compass(hor, mag.az, "2016/04/02")
```

---

reduct.theodolite *Data reduction for theodolite measurements using the sun-sight method*

Description

This function calculates the true azimuth of a structure measured with a theodolite using the sunsight technique.

Usage

```
reduct.theodolite(loc, az, date, time, tz, az.sun = 0, alt, name, ID, HWT.ID)
```

Arguments

loc	Location, either a *skyscapeR.object* or a vector containing the latitude and longitude of location, in this order.
az	Array of azimuths. Use ten to convert to decimal point format if necessary.
date	Date of measurements as a string in the format: 'YYYY/MM/DD'
time	Time of sun-sight measurement in the format: 'HH:MM:SS'
tz	Timezone of input wither as a known acronym (eg. "GMT", "CET") or a string with continent followed by country capital (eg. "Europe/London").
az.sun	(Optional) Measured azimuth of the sun. Defaults to zero.

alt	(Optional) Altitude, necessary for automatic declination calculation. If missing and *loc* is a *skyscapeR.horizon* object then the altitude will be automatically read from the horizon profile.
name	(Optional) Names or labels to identify each measurement.
ID	(Optional) IDs or codes to identify each measurement.
HWT.ID	(Optional) HeyWhatsThat IDs relating to a previously generated horizon profile for measurement.

References

Ruggles, C.L.N. (1999). *Astronomy in Prehistoric Britain and Ireland*. Yale University Press.

See Also

sunAz, ten, sixty

Examples

```
lat <- ten(35,50,37.8)
lon <- ten(14,34,6.4)
az <- c( ten(298,24,10), ten(302,20,40))
az.sun <- ten(327,29,50)
date <- "2016/02/20"
time <- "11:07:17"

data <- reduct.theodolite(c(lat,lon), az, date , time, tz= "Europe/Malta", az.sun)

# Declination will be automatically calculated if the altitude is also given:
data <- reduct.theodolite(c(lat,lon), az, date , time, tz= "Europe/Malta", az.sun, alt=c(2,5))

# Alternatively, the altitude can be automatically retrieved from a horizon profile:
hor <- download.HWT('UFXERSLQ')
data <- reduct.theodolite(hor, az, date, time, tz= "Europe/Malta", az.sun)
```

RugglesCKR *Cork and Kerry Stone Row Data*

Description

Data from C.L.N. Ruggles' fieldwork on the Stone Rows of Cork and Kerry.

Usage

```
data(RugglesCKR)
```

Format

A data frame with 41 rows and 5 variables:

Ref Site Ref

NE.SW String indicating whether they are towards the NE or SW

Az.Hill Azimuth of hill towards which the Stone Rows are pointing

Alt.Hill Altitude of hill towards which the Stone Rows are pointing

Dec.Hill Declination of hill towards which the Stone Rows are pointing

References

Ruggles, C.L.N. (1999). *Astronomy in Prehistoric Britain and Ireland*. Yale University Press.

Examples

```
data(RugglesCKR)
curv <- curvigram(RugglesCKR$Dec.Hill, 2)
plotCurv(curv)
```

RugglesRSC *Recumbent Stone Circle Data*

Description

Declination data from C.L.N. Ruggles' fieldwork on the Scottish Recumbent Stone Circles.

Usage

```
data(RugglesRSC)
```

Format

A data frame with 37 rows and 2 variables:

Dec Declination

ID Site ID

References

Ruggles, C.L.N. (1999). *Astronomy in Prehistoric Britain and Ireland*. Yale University Press.

Examples

```
data(RugglesRSC)
curv <- curvigram(RugglesRSC$Dec, 2)
plotCurv(curv)
```

`sigTest` *Perform a null hypothesis significance test of a given curvigram*

Description

This function performs a null hypothesis significance test, for a given curvigram and null hypothesis and outputs a p-value as well as all the information needed for ancillary plotting.

Usage

```
sigTest(curv, null.hyp, level = 0.95, type = "2-tailed", nsims = 2000,
  ncores)
```

Arguments

`curv`	Object of *skyscapeR.curv* format, created using `curvigram`
`null.hyp`	Object of *skyscapeR.nh* format, created with one of the Null Hypothesis models of *skyscapeR* (see See Also section below).
`level`	(Optional) Level of confidence for p-value calculation and output. Defaults to 0.95, i.e. a 95% confidence envelope.
`type`	(Optional) Whether the test is to be '1-tailed' or '2-tailed'. Defaults to '2-tailed'.
`nsims`	(Optional) Number of simulations to run. The higher this number the slower this process will be, but the lower it is the less power the method has. Defaults to 2000 as a base minimum to test for significance at the p=0.0005 level, but the recommended value is 10,000.
`ncores`	(Optional) Number of processing cores to use for parallelisation. Defaults to the number of available cores minus 1.

See Also

`nh.Uniform`, `nh.SummerFM`, `plotCurv`, `plotZscore`

Examples

```
## Not run:
data(RugglesRSC)
curv <- curvigram(RugglesRSC$Dec, sd=2)
sig <- sigTest(curv, null.hyp=nh.Uniform(c(57,2)))

plotCurv(curv, signif=sig)

## End(Not run)
```

`sky.objects` *Creates a* skyscapeR.object *for plotting of celestial objects at given epoch*

Description

This function creates an object containing all the necessary information to plot celestial objects/events unto the many plotting functions of *skyscapeR* package.

Usage

```
sky.objects(names, epoch, col = "red", lty = 1, lwd = 1)
```

Arguments

`names`	The name(s) of the celestial object(s) or event(s) of interest. These can be one of the following soli-lunar events: *jS*, *dS*, *eq*, *nmnLX*, *nMjLX*, *smnLX*, *sMjLX*, or the name of any star in the database. As shorthand, the names *sun* and *moon* can be used to represent all the above solar and lunar events, respectively. Alternatively, custom declination values can also be used.
`epoch`	The year or year range (as an array) one is interested in.
`col`	(Optional) The colour for plotting, and differentiating these objects. Defaults to red for all objects.
`lty`	(Optional) Line type (see `par`) used for differentiation. Only activated for single year epochs.
`lwd`	(Optional) Line width (see `par`) used for differentiation. Only activated for single year epochs.

Examples

```
## Not run:
# Create a object with solar targets for epoch range 4000-2000 BC:
tt <- sky.objects('sun', c(-4000,-2000))

# Create an object with a few stars for same epoch:
tt <- sky.objects(c('Sirius', 'Betelgeuse', 'Antares'), c(-4000,-2000),
col=c('white', 'red', 'orange'))

# Create an object with solstices and a custom declination value:
tt <- sky.objects(c('dS','jS', -13), c(-4000,-2000))

## End(Not run)
```

sMjLX — *Declination of southern major Lunar Extreme for a given year*

Description

This function calculates the declination of the southern major Lunar Extreme for a given year, by simple addition of obliquity with maximum lunar inclination.

Usage

```
sMjLX(year = cur.year)
```

Arguments

year	Year for which to calculate the declination. Defaults to present year as given by *Sys.Date()*.

See Also

nmnLX, nMjLX, smnLX

Examples

```
# Southern major Lunar Extreme declination for year 2499 BC:
sMjLX(-2500)
```

smnLX — *Declination of southern minor Lunar Extreme for a given year*

Description

This function calculates the declination of the southern minor Lunar Extreme for a given year, by simple addition of obliquity with maximum lunar inclination.

Usage

```
smnLX(year = cur.year)
```

Arguments

year	Year for which to calculate the declination. Defaults to present year as given by *Sys.Date()*.

See Also

nmnLX, nMjLX, sMjLX

Examples

```
# Southern minor Lunar Extreme declination for year 2499 BC:
smnLX(-2500)
```

`star` *Create* skyscapeR.star *object*

Description

This function retrieves information for a given star and saves it in the *skyscapeR.star* format ready to be used by other skyscapeR package function.

Usage

```
star(string, year)
```

Arguments

`string`	This can be either: 1) the common name for the star, 2) the Hipparchos catalogue number (if string begins with HIP), 3) the Bright Star Catalogue number (if string begins with HR), 4) the Messier catalogue number (if string begins with M), or 5) the Bayer designation (if string has exactly 5 characters, the second of which is a space).
`year`	Year for which to calculate the coordinates. Defaults to J2000.0 epoch.

See Also

`precess`, `co_nutate`, `co_aberration`

Examples

```
# Retrieve data for Aldebaran:
Aldeb <- star('Aldebaran')

# Retrieve data for the Pleiades (M45):
P1 <- star('Pleiades')
P2 <- star('M45')

# Retrieve data for Sirius on 2999 BC:
ss <- star('Sirius', -3000)
```

`star.phases` *Calculate the seasons and phases of a star*

Description

This function calculates the seasons (Rising, Setting, etc.) and phases (Arising and Lying Hidden, Curtailed Passage) of a star for a given location and epoch. This functions uses the *arcus visionis* approximation of Purrington (1988). For the nomenclature used, and description of star phases, see Brady (2015).

Usage

```
star.phases(star, year, loc, alt.hor = 0, alt.rs = 10, res = 24/3600,
  ncores)
```

Arguments

`star`	Either the star name or a *skyscapeR.star* object.
`year`	The year of interest.
`loc`	Location, either a *skyscapeR.object* or a vector containing the latitude and longitude of location, in this order.
`alt.hor`	(Optional) The altitude of the horizon to consider. Defaults to zero degrees.
`alt.rs`	(Optional) The maximum altitude of a star's first or last visibility for it to still be considered to be as rising or setting. Defaults to ten degrees.
`res`	(Optional) Resolution of calculation. The smaller this figure the slower the computation. Defaults to 24/3600 = 1 sec.
`ncores`	(Optional) Number of processing cores to use for parallelisation. Defaults to the number of available cores minus 1.

References

Purrington, Robert D. (1988) Heliacal Rising and Setting: Quantitative Aspects, *Journal for the History of Astronomy (Archaeoastronomy Supplement 12)* 19, S72-S84. Available online at [SAO/NASA ADS Astronomy Abstract Service](http://adsabs.harvard.edu/abs/1988JHAS...19...72P)

Brady, Bernadette (2015) Star Phases: the Naked-eye Astronomy of the Old Kingdom Pyramid Texts. In F Silva and N Campion (eds) *Skyscapes: The Role and Importance of the Sky in Archaeology*. Oxford: Oxbow Books, pp. 76-86.

See Also

`plotPhases`

Examples

```
## Not run:
ss1 <- star.phases('Aldebaran',-4000, c(35,-8))

# One can then look at the star's phase:
ss1$phase

# Date range of seasons:
ss1$seasons

# Date range of events/phases:
ss1$events

# And plot them:
plotPhases(ss1)

# You can play with the parameters and see how predictions change:
ss1 <- star.phases('Aldebaran',-4000, c(35,-8), alt.hor=2, alt.rs=5)
plotPhases(ss1)

## End(Not run)
```

---

stars *Stellar Data*

Description

Data on about one hundred brightest stars and clusters taken from the *Bright Star Catalog, 5th Edition*. It includes Name, Constellation, Bayer Designation, Apparent Magnitude, Right Ascension, Declination, Proper Motion (in both RA and Dec) and Colour. Coordinates are given for J2000 epoch.

Usage

```
data(stars)
```

Format

A data frame with 103 rows and 13 variables:

HIP.ID Hipparchus Catalogue ID

HR.ID Bright Star Catalogue ID

MESSIER.ID Messier Catalogue ID

BAYER.DESIGNATION Bayer Designation

BAYER Bayer label

CONSTELLATION Constellation star belongs to

NAME Common Name

VMAG Apparent Magnitude

RA Right Ascension

DEC Declination

PM_RA Proper Motion in RA

PM_DEC Proper Motion in DEC

COLOUR Colour

References

Hoffleit, D.; Jaschek, C. (1991). *The Bright star catalogue*. New Haven: Yale University Observatory. (Bright Star Catalogue)

Examples

```
data(stars)
stars$VMAG[stars$NAME == 'Sirius']
```

---

sunAz *Returns the azimuth of the sun at a given time from a specific location*

Description

This function returns the azimuth of the sun at a given time and location, useful for data reduction of theodolite measurements using the sunsight technique (`reduct.theodolite`).

Usage

```
sunAz(loc, time, timezone)
```

Arguments

`loc`	Location, either a *skyscapeR.object* or a vector containing the latitude and longitude of location, in this order.
`time`	String containing the date and time in the following format: "YYYY-MM-DD HH:MM:SS"
`timezone`	Timezone of input either as a known acronym (eg. "GMT", "CET") or a string with continent followed by country capital (eg. "Europe/London").

See Also

`reduct.theodolite`

Examples

```
sunAz(c(52,-3), '2017-10-04 12:32:14', 'Europe/London')
```

`zenith` *Declination of the zenith sun for a given location*

Description

This function returns the declination of the sun when it is at the zenith for a given location. If this phenomena does not occur at given location (i.e. if location is outside the tropical band) the function returns a *NULL* value.

Usage

```
zenith(loc)
```

Arguments

`loc`	This can be either the latitude of the location, or a *skyscapeR.horizon* object.

See Also

`jS`, `dS`, `eq`, `antizenith`

Examples

```
# Zenith sun declination for Mexico City:
zenith(19.419)

# There is no zenith sun phenomena in London:
zenith(51.507)
```

Index