Package: suntools (via r-universe)

October 27, 2024

Version 1.0.0

Date 2023-06-08

Title Calculate Sun Position, Sunrise, Sunset, Solar Noon and Twilight

Encoding UTF-8

Depends R (>= 2.10)

Imports methods, sf, stats

Description Provides a set of convenient functions for calculating sun-related information, including the sun's position (elevation and azimuth), and the times of sunrise, sunset, solar noon, and twilight for any given geographical location on Earth. These calculations are based on equations provided by the National Oceanic & Atmospheric Administration (NOAA) <https://gml.noaa.gov/grad/solcalc/calcdetails.html> as described in ``Astronomical Algorithms'' by Jean Meeus (1991, ISBN: 978-0-943396-35-4). A resource for researchers and professionals working in fields such as climatology, biology, and renewable energy.

License GPL (>= 3)

URL https://github.com/adokter/suntools/

BugReports https://github.com/adokter/suntools/issues

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

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

RemoteUrl https://github.com/adokter/suntools

RemoteRef HEAD

RemoteSha 374fff7b2f595fabb8bf75d928c63b6e2f099e98

Contents

---

crepuscule *Compute crepuscular time*

Description

Calculates the crepuscular time, i.e., the time of dawn or dusk at a specific geographical location and time.

Usage

```
crepuscule(crds, dateTime, ...)

## S4 method for signature 'sf,POSIXct'
crepuscule(
  crds,
  dateTime,
  solarDep,
  direction = c("dawn", "dusk"),
  POSIXct.out = FALSE
)

## S4 method for signature 'matrix,POSIXct'
crepuscule(
  crds,
  dateTime,
  crs = sf::st_crs(4326),
  solarDep,
  direction = c("dawn", "dusk"),
  POSIXct.out = FALSE
)

## S4 method for signature 'SpatialPoints,POSIXct'
crepuscule(
  crds,
  dateTime,
  solarDep,
  direction = c("dawn", "dusk"),
  POSIXct.out = FALSE
)
```

Arguments

`crds`	Geographical coordinates. It can be an object of class `sf`, `matrix`, or `SpatialPoints`.
`dateTime`	A `POSIXct` object representing the date and time.
`...`	Additional arguments that are passed to methods.
`solarDep`	A numerical value representing the solar depression angle.
`direction`	Character, determines whether to calculate the time of sunrise or sunset.
`POSIXct.out`	Logical, if `TRUE`, the result is returned as a `POSIXct` object, otherwise, it is returned as a fraction of a day.
`crs`	A `CRS` object representing the coordinate reference system. Default is `sf::st_crs(4326)` which denotes WGS84 (World Geodetic System 1984).

Details

Methods are available for different classes of geographical coordinates, including:

- `sf`: an object of class `sf`.
- `matrix`: An unnamed matrix of coordinates, with each row containing a pair of geographical coordinates in `c(lon, lat)` order. See the example below.
- `SpatialPoints`: an object of class `SpatialPoints`. Input can consist of one location and at least one `POSIXct` time, or one `POSIXct` time and at least one location. `solarDep` is recycled as needed. Do not use the daylight savings time zone string for supplying `dateTime`, as many OS will not be able to properly set it to standard time when needed.

Compared to NOAA's original Javascript code, the sunrise and sunset estimates from this translation may differ by +/- 1 minute, based on tests using selected locations spanning the globe. This translation does not include calculation of prior or next sunrises/sunsets for locations above the Arctic Circle or below the Antarctic Circle.

Solar position calculation:

Details for the calculations are provided by NOAA here, which we repeat below as a reference.

The calculations in the NOAA Sunrise/Sunset and Solar Position Calculators are based on equations from Astronomical Algorithms, by Jean Meeus. The sunrise and sunset results are theoretically accurate to within a minute for locations between +/- 72° latitude, and within 10 minutes outside of those latitudes. However, due to variations in atmospheric composition, temperature, pressure and conditions, observed values may vary from calculations.

For the purposes of these calculators the current Gregorian calendar is extrapolated backward through time. When using a date before 15 October, 1582, you will need to correct for this. The year preceding year 1 in the calendar is year zero (0). The year before that is -1. The approximations used in these programs are very good for years between 1800 and 2100. Results should still be sufficiently accurate for the range from -1000 to 3000. Outside of this range, results may be given, but the potential for error is higher.

Atmospheric refraction correction:

For sunrise and sunset calculations, we assume 0.833° of atmospheric refraction. In the solar position calculator, atmospheric refraction is modeled as:

Solar Elevation	Approximate Atmospheric Refraction Correction (°)
85° to 90°	0
5° to 85°	$\frac{1}{3600}\left(\frac{58.1}{\tan(h)} - \frac{0.07}{\tan^3(h)} + \frac{0.000086}{\tan^5(h)}\right)$
-0.575° to 5°	$\frac{1}{3600}\left(1735 - 518.2h + 103.4h^2 - 12.79h^3 + 0.711h^4\right)$
< -0.575°	$\frac{1}{3600}\left(\frac{-20.774}{\tan(h)}\right)$

The effects of the atmosphere vary with atmospheric pressure, humidity and other variables. Therefore the solar position calculations presented here are approximate. Errors in sunrise and sunset times can be expected to increase the further away you are from the equator, because the sun rises and sets at a very shallow angle. Small variations in the atmosphere can have a larger effect.

Value

data.frame with the time of crepuscular light as a fraction of a day; if POSIXct.out=TRUE returns an additional POSIXct timestamp column (default = TRUE)

References

- Meeus, J. (1991) Astronomical Algorithms. Willmann-Bell, Inc.
- NOAA's sunrise/sunset calculator. These algorithms include corrections for atmospheric refraction effects.
- NOAA's solar calculations details

Examples

```
#Civil dawn in Ithaca, NY on June 1, 2023

crepuscule(
       matrix(c(-76.4511, 42.4800), nrow = 1),
       as.POSIXct("2023-06-01", tz = "America/New_York"),
       solarDep = 6,
       direction = "dawn",
       POSIXct.out = TRUE
     )
```

---

solarnoon *Compute solar noon time*

Description

Calculates the solar noon, i.e., the time when the sun is at its highest point in the sky at a specific geographical location and time.

Usage

```
solarnoon(crds, dateTime, ...)

## S4 method for signature 'sf,POSIXct'
solarnoon(crds, dateTime, POSIXct.out = FALSE)

## S4 method for signature 'matrix,POSIXct'
solarnoon(crds, dateTime, crs = sf::st_crs(4326), POSIXct.out = FALSE)

## S4 method for signature 'SpatialPoints,POSIXct'
solarnoon(crds, dateTime, POSIXct.out = FALSE)
```

Arguments

crds	Geographical coordinates. It can be an object of class `sf`, `matrix`, or `SpatialPoints`.
dateTime	A `POSIXct` object representing the date and time.
...	Additional arguments that are passed to methods.
POSIXct.out	Logical, if `TRUE`, the result is returned as a `POSIXct` object, otherwise, it is returned as a fraction of a day.
crs	A `CRS` object representing the coordinate reference system. Default is `sf::st_crs(4326)` which denotes WGS84 (World Geodetic System 1984).

Details

Methods are available for different classes of geographical coordinates, including:

- `sf`: an object of class `sf`.
- `matrix`: An unnamed matrix of coordinates, with each row containing a pair of geographical coordinates in `c(lon, lat)` order. See the example below.
- `SpatialPoints`: an object of class `SpatialPoints`. Input can consist of one location and at least one `POSIXct` time, or one `POSIXct` time and at least one location. `solarDep` is recycled as needed. Do not use the daylight savings time zone string for supplying `dateTime`, as many OS will not be able to properly set it to standard time when needed.

Compared to NOAA's original Javascript code, the sunrise and sunset estimates from this translation may differ by +/- 1 minute, based on tests using selected locations spanning the globe. This translation does not include calculation of prior or next sunrises/sunsets for locations above the Arctic Circle or below the Antarctic Circle.

Solar position calculation:

Details for the calculations are provided by NOAA here, which we repeat below as a reference.

The calculations in the NOAA Sunrise/Sunset and Solar Position Calculators are based on equations from Astronomical Algorithms, by Jean Meeus. The sunrise and sunset results are theoretically accurate to within a minute for locations between +/- 72° latitude, and within 10 minutes outside of those latitudes. However, due to variations in atmospheric composition, temperature, pressure and conditions, observed values may vary from calculations.

For the purposes of these calculators the current Gregorian calendar is extrapolated backward through time. When using a date before 15 October, 1582, you will need to correct for this. The year preceding year 1 in the calendar is year zero (0). The year before that is -1. The approximations used in these programs are very good for years between 1800 and 2100. Results should still be sufficiently accurate for the range from -1000 to 3000. Outside of this range, results may be given, but the potential for error is higher.

Atmospheric refraction correction:

For sunrise and sunset calculations, we assume 0.833° of atmospheric refraction. In the solar position calculator, atmospheric refraction is modeled as:

Solar Elevation	Approximate Atmospheric Refraction Correction (°)
85° to 90°	0
5° to 85°	$\frac{1}{3600}\left(\frac{58.1}{\tan(h)} - \frac{0.07}{\tan^3(h)} + \frac{0.000086}{\tan^5(h)}\right)$
-0.575° to 5°	$\frac{1}{3600}\left(1735 - 518.2h + 103.4h^2 - 12.79h^3 + 0.711h^4\right)$
< -0.575°	$\frac{1}{3600}\left(\frac{-20.774}{\tan(h)}\right)$

The effects of the atmosphere vary with atmospheric pressure, humidity and other variables. Therefore the solar position calculations presented here are approximate. Errors in sunrise and sunset times can be expected to increase the further away you are from the equator, because the sun rises and sets at a very shallow angle. Small variations in the atmosphere can have a larger effect.

Value

data.frame with the time of solar noon as a fraction of a day; if POSIXct.out=TRUE returns an additional POSIXct timestamp column (default = TRUE)

References

- Meeus, J. (1991) Astronomical Algorithms. Willmann-Bell, Inc.
- NOAA's solar position calculator. These algorithms include corrections for atmospheric refraction effects.
- NOAA's solar calculations details

Examples

```
# Solar noon in Ithaca, NY, USA on June 1, 2023

solarnoon(
 matrix(c(-76.4511, 42.4800), nrow = 1),
 as.POSIXct("2023-06-01", tz = "America/New_York"),
 POSIXct.out=TRUE
)
```

solarpos *Compute solar position*

Description

Calculates the solar position, i.e., the sun's elevation and azimuth, at a specific geographical location and time.

Usage

```
solarpos(crds, dateTime, ...)

## S4 method for signature 'sf,POSIXct'
solarpos(crds, dateTime, ...)

## S4 method for signature 'matrix,POSIXct'
solarpos(crds, dateTime, crs = sf::st_crs(4326), ...)

## S4 method for signature 'SpatialPoints,POSIXct'
solarpos(crds, dateTime, ...)
```

Arguments

`crds`	Geographical coordinates. It can be an object of class `sf`, `matrix`, or `SpatialPoints`.
`dateTime`	A `POSIXct` object representing the date and time.
`...`	Additional arguments that are passed to methods.
`crs`	A `CRS` object representing the coordinate reference system. Default is `sf::st_crs(4326)`.

Details

Methods are available for different classes of geographical coordinates, including:

- `sf`: an object of class `sf`.
- `matrix`: An unnamed matrix of coordinates, with each row containing a pair of geographical coordinates in `c(lon, lat)` order. See the example below.
- `SpatialPoints`: an object of class `SpatialPoints`. Input can consist of one location and at least one `POSIXct` time, or one `POSIXct` time and at least one location. `solarDep` is recycled as needed. Do not use the daylight savings time zone string for supplying `dateTime`, as many OS will not be able to properly set it to standard time when needed.

Compared to NOAA's original Javascript code, the sunrise and sunset estimates from this translation may differ by +/- 1 minute, based on tests using selected locations spanning the globe. This translation does not include calculation of prior or next sunrises/sunsets for locations above the Arctic Circle or below the Antarctic Circle.

Solar position calculation:

Details for the calculations are provided by NOAA here, which we repeat below as a reference.

The calculations in the NOAA Sunrise/Sunset and Solar Position Calculators are based on equations from Astronomical Algorithms, by Jean Meeus. The sunrise and sunset results are theoretically accurate to within a minute for locations between +/- 72° latitude, and within 10 minutes outside of those latitudes. However, due to variations in atmospheric composition, temperature, pressure and conditions, observed values may vary from calculations.

For the purposes of these calculators the current Gregorian calendar is extrapolated backward through time. When using a date before 15 October, 1582, you will need to correct for this. The year preceding year 1 in the calendar is year zero (0). The year before that is -1. The approximations used in these programs are very good for years between 1800 and 2100. Results should still be sufficiently accurate for the range from -1000 to 3000. Outside of this range, results may be given, but the potential for error is higher.

Atmospheric refraction correction:

For sunrise and sunset calculations, we assume 0.833° of atmospheric refraction. In the solar position calculator, atmospheric refraction is modeled as:

Solar Elevation	Approximate Atmospheric Refraction Correction (°)
85° to 90°	0
5° to 85°	$\frac{1}{3600}\left(\frac{58.1}{\tan(h)} - \frac{0.07}{\tan^3(h)} + \frac{0.000086}{\tan^5(h)}\right)$
-0.575° to 5°	$\frac{1}{3600}\left(1735 - 518.2h + 103.4h^2 - 12.79h^3 + 0.711h^4\right)$
< -0.575°	$\frac{1}{3600}\left(\frac{-20.774}{\tan(h)}\right)$

The effects of the atmosphere vary with atmospheric pressure, humidity and other variables. Therefore the solar position calculations presented here are approximate. Errors in sunrise and sunset times can be expected to increase the further away you are from the equator, because the sun rises and sets at a very shallow angle. Small variations in the atmosphere can have a larger effect.

Value

matrix with the solar azimuth (in degrees from North), and elevation.

References

- Meeus, J. (1991) Astronomical Algorithms. Willmann-Bell, Inc.
- NOAA's solar position calculator. These algorithms include corrections for atmospheric refraction effects.
- NOAA's solar calculations details

Examples

```
# Solar position in Ithaca, NY, USA on June 1, 2023 at 08:00:00

solarpos(
```

```
 matrix(c(-76.4511, 42.4800), nrow = 1),
 as.POSIXct("2023-06-01 08:00:00", tz = "America/New_York")
)
```

---

sunriset *Calculate sunrise/sunset*

Description

Calculates sunrise or sunset at a specific geographical location and time depending on the `direction` parameter.

Usage

```
sunriset(crds, dateTime, ...)

## S4 method for signature 'sf,POSIXct'
sunriset(
  crds,
  dateTime,
  direction = c("sunrise", "sunset"),
  POSIXct.out = FALSE
)

## S4 method for signature 'matrix,POSIXct'
sunriset(
  crds,
  dateTime,
  crs = sf::st_crs(4326),
  direction = c("sunrise", "sunset"),
  POSIXct.out = FALSE
)

## S4 method for signature 'SpatialPoints,POSIXct'
sunriset(
  crds,
  dateTime,
  direction = c("sunrise", "sunset"),
  POSIXct.out = FALSE
)
```

Arguments

`crds`	Geographical coordinates. It can be an object of class `sf`, `matrix`, or `SpatialPoints`.
`dateTime`	A `POSIXct` object representing the date and time.
`...`	Additional arguments that are passed to methods.
`direction`	Character, determines whether to calculate the time of sunrise or sunset.

`POSIXct.out`	Logical, if `TRUE`, the result is returned as a `POSIXct` object, otherwise, it is returned as a fraction of a day.
`crs`	A "CRS" object representing the coordinate reference system. Default is `sf::st_crs(4326)` which denotes WGS84 (World Geodetic System 1984).

Details

Methods are available for different classes of geographical coordinates, including:

- `sf`: an object of class `sf`.
- `matrix`: An unnamed matrix of coordinates, with each row containing a pair of geographical coordinates in `c(lon, lat)` order. See the example below.
- `SpatialPoints`: an object of class `SpatialPoints`. Input can consist of one location and at least one `POSIXct` time, or one `POSIXct` time and at least one location. `solarDep` is recycled as needed. Do not use the daylight savings time zone string for supplying `dateTime`, as many OS will not be able to properly set it to standard time when needed.

Compared to NOAA's original Javascript code, the sunrise and sunset estimates from this translation may differ by +/- 1 minute, based on tests using selected locations spanning the globe. This translation does not include calculation of prior or next sunrises/sunsets for locations above the Arctic Circle or below the Antarctic Circle.

Solar position calculation:

Details for the calculations are provided by NOAA here, which we repeat below as a reference.

The calculations in the NOAA Sunrise/Sunset and Solar Position Calculators are based on equations from Astronomical Algorithms, by Jean Meeus. The sunrise and sunset results are theoretically accurate to within a minute for locations between +/- 72° latitude, and within 10 minutes outside of those latitudes. However, due to variations in atmospheric composition, temperature, pressure and conditions, observed values may vary from calculations.

For the purposes of these calculators the current Gregorian calendar is extrapolated backward through time. When using a date before 15 October, 1582, you will need to correct for this. The year preceding year 1 in the calendar is year zero (0). The year before that is -1. The approximations used in these programs are very good for years between 1800 and 2100. Results should still be sufficiently accurate for the range from -1000 to 3000. Outside of this range, results may be given, but the potential for error is higher.

Atmospheric refraction correction:

For sunrise and sunset calculations, we assume 0.833° of atmospheric refraction. In the solar position calculator, atmospheric refraction is modeled as:

Solar Elevation	Approximate Atmospheric Refraction Correction (°)
85° to 90°	0
5° to 85°	$\frac{1}{3600}\left(\frac{58.1}{\tan(h)} - \frac{0.07}{\tan^3(h)} + \frac{0.000086}{\tan^5(h)}\right)$
-0.575° to 5°	$\frac{1}{3600}\left(1735 - 518.2h + 103.4h^2 - 12.79h^3 + 0.711h^4\right)$
< -0.575°	$\frac{1}{3600}\left(\frac{-20.774}{\tan(h)}\right)$

The effects of the atmosphere vary with atmospheric pressure, humidity and other variables. Therefore the solar position calculations presented here are approximate. Errors in sunrise and sunset times can be expected to increase the further away you are from the equator, because the sun rises and sets at a very shallow angle. Small variations in the atmosphere can have a larger effect.

Value

data.frame with the time of sunrise as a fraction of a day; if POSIXct.out=TRUE returns an additional POSIXct timestamp column (default = TRUE)

References

- Meeus, J. (1991) Astronomical Algorithms. Willmann-Bell, Inc.
- NOAA's sunrise/sunset calculator. These algorithms include corrections for atmospheric refraction effects.
- NOAA's solar calculations details

Examples

```
#Sunset in Ithaca, NY, USA on June 1, 2023

sunriset(
 matrix(c(-76.4511, 42.4800), nrow = 1),
 as.POSIXct("2023-06-01", tz = "America/New_York"),
 direction='sunset',
 POSIXct.out=TRUE
)
```

Index