Package ‘astroFns’

October 12, 2022

Type  Package
Title  Astronomy: Time and Position Functions, Misc. Utilities
Version 4.2-1
Date  2022-05-08
Author Andrew Harris
Maintainer Andrew Harris <harris@astro.umd.edu>
Description Miscellaneous astronomy functions, utilities, and data.
License GPL (>= 2)
LazyLoad yes
NeedsCompilation no
Repository CRAN
Date/Publication 2022-05-08 17:20:02 UTC

R topics documented:

astroFns-package ......................................................... 2
angSep ................................................................. 2
b2j ................................................................. 3
beamDiskOverlap ..................................................... 4
dmjd2ut ............................................................. 5
dms2rad ............................................................. 6
elev ................................................................. 7
gmst1 .............................................................. 9
hms2rad ........................................................... 10
j2b ............................................................... 11
jd2ymd ............................................................ 12
planetFlux .......................................................... 13
rad2dms ........................................................... 14
rad2hms ........................................................... 15
ut2dmjd ........................................................... 15
ut2lst ............................................................ 16
ymd2jd ............................................................ 18
Description

Collection of time, position, and utility functions for astronomy: Julian and Modified Julian Day transformations, J2000/B1950 coordinate transformations, GMST at 0h from UT, UT to LST and hour angle, angular unit transformations and distance, etc. Precision generally at the level of a millisecond in time for JD, 0.1 s for LST, and a few tenths of an arcsecond in position for B2000-J1950 and vice versa. Additional functions include flux from thermal disk-shaped source, and others. Functions were originally assembled to support single-dish radio astronomy planning and observations. Cosmology-related functions are in the cosmoFns package.

Details

Package: astroFns
Type: Package
Version: 4.2-0
Date: 2022-05-08
License: GPL (>=2)
LazyLoad: yes

Author(s)

Andrew Harris Maintainer: Andrew Harris <harris@astro.umd.edu>

angSep

Angular separation of two sky positions

Description

angSep calculates the angular separation of two sky positions using spherical trigonometry.

Usage

angSep(ra1, dec1, ra2, dec2)
b2j

Arguments

ra1 Right ascension (string) of the first position.
dec1 Declination of (string) the first position.
ra2 Right ascension (string) the second position.
dec2 Declination of (string) the second position.

Details

Enter positions as text strings with fields separated by characters d, h, m, s, a colon, or a comma, e.g. '17, 42, 28', '-28h43m03s', or '- 28 :43 : 3'. Spaces are removed in input conversion. This is a spherical trigonometry calculation, valid for small and large distances.

Value

Returns angular separation in decimal degrees.

Author(s)

Andrew Harris

See Also

See dms2rad, hms2rad for input conversions.

Examples

angSep("1, 59, 03", "-3, 40, 44", "2, 30", "5, 40, 03")
angSep("1h59m03s", "-3d40m44s", "2h30", "5h40m03")
angSep("1", "0", "2", "0")
angSep(" 1, 40, 4", " - 5, 6", "3", "1")

b2j  
B1950 to J2000 coordinate conversion

Description

Precession from B1950 to J2000

Usage

b2j(ra = "17h42m29.3076s", dec = "-28d59m18.484s")

Arguments

ra B1950 Right ascension (string)
dec B1950 Declination (string)
beamDiskOverlap

**Details**

Enter positions as text strings with fields separated by characters d, h, m, s, a colon, or a comma, e.g. '17, 42, 28', '-28h43m03s', or '- 28 :43 : 3'. Spaces are removed in input conversion. Trailing missing values are taken as zero. The code uses an approximate formula for precession; spot checks give results accurate within a few tenths of an arcsecond.

**Value**

List with strings in:

- `ra2000` J2000 Right ascension
- `dec2000` J2000 Declination

**Note**

Calculation based on power-law expansion of exact function.

**Author(s)**

Andrew Harris

**References**


**See Also**

`j2b`. See `dms2rad`, `hms2rad` for input conversions.

**Examples**

```r
b2j()
b2j(ra='17, 43', dec='28, 47, 30')
b2j(ra='17, 43', dec='28 - 28, 47, 30')
b2j(ra='17h43m', dec='28d47m30s')
tmp <- b2j(ra='17, 43', dec=' 28, 47, 30')
str(tmp)
tmp
```

beamDiskOverlap  

**Gaussian beam and disk overlap with shift**

**Description**

Calculate the overlap integral of a 2-D Gaussian beam and a uniform disk, including a shift between the centers of the beam and disk.
**Usage**

beamDiskOverlap(s = 0, r = 1, theta.fwhm = 1)

**Arguments**

- **s**: Shift between centers
- **r**: Disk radius
- **theta.fwhm**: Gaussian beam FWHM

**Details**

Converts the 2-D integral to 1-D for speed. Use consistent units.

**Value**

Value of the overlap integral, normalized to unity for a beam much smaller than the disk.

**Author(s)**

Andrew Harris

**References**

"Telescope illumination and beam measurements for submillimeter astronomy," A.I. Harris, Internat. J. IR and mm Waves, 9, 231 (1988)

**Examples**

```r
s <- seq(0, 10, 0.1)
plot(s, beamDiskOverlap(s, 4, 1), t='l', col=4)
```

---

**dmjd2ut**

*DMJD to UT*

**Description**

Decimal modified Julian date to Universal time.

**Usage**

dmjd2ut(dmjd, tz='UTC')

**Arguments**

- **dmjd**: Time in decimal Modified Julian Date
- **tz**: Time zone string
**dms2rad**

**Details**

Calculation is always from UTC, but it is possible to correct to local time zone with tz (see Sys.timezone). For instance, tz = 'EST5EDT' converts to U.S. Eastern time, with EST or EDT based on the system’s knowledge of the date for switching between the two. Set the number of digits after the decimal place for seconds, n, with options('digits.secs'=n).

**Value**

Time string with class POSIXct

**Author(s)**

Andrew Harris

**See Also**

ut2dmjd, ymd2jd, strptime, ISOdatetime, axis.POSIXct for time in plot axes; as.POSIXct to recover time in plot from locator()

**Examples**

```r
dmjd2ut(56951.54183613)
sd <- getOption('digits.secs')
dmjd2ut(ut2dmjd(2010, 1, 5, 2, 34, 17.8115))
options('digits.secs' = 3)
dmjd2ut(ut2dmjd(2015, 1, 5, 2, 34, 17.8115))
options('digits.secs' = sd)
dmjd2ut(ut2dmjd(2015, 1, 5, 2, 34, 17.8115), tz='CET')
dmjd2ut(ut2dmjd(2015, 8, 5, 2, 34, 17.8115), tz='CET')
dmjd2ut(ut2dmjd(2015, 1, 5, 2, 34, 17.8115), tz='EST5EDT')
dmjd2ut(ut2dmjd(2015, 8, 5, 2, 34, 17.8115), tz='EST5EDT')
dmjd2ut(ymd2jd(2001, 1, 1) - 2400000.5)
```

---

**dms2rad**

*Degrees, minutes, and seconds to radians*

**Description**

Angular conversion from degrees, minutes, and seconds to radians

**Usage**

```r
dms2rad(d = '33d 09m 35.0s')
```
**Arguments**

- **d**: String containing degrees, minutes, and seconds

**Details**

Function reads a string (the input is a string to allow conversion of angles between -1 and zero degrees) with degrees, minutes, and seconds separated by any of characters d, m, s, a colon, or a comma. Spaces are not valid separators, as they are removed as part of input parsing. Decimal values are allowed in any position. Zeros are the default if values for minutes or seconds are missing from the string. A minus sign, W, or w before the degrees indicates negative degrees. Positive degrees are denoted by no character, +, E, or e before the degrees values.

**Value**

Angle in radians

**Author(s)**

Andrew Harris

**See Also**

* hms2rad, rad2dms, rad2hms

**Examples**

```r
dms2rad('10, 22, 14')
dms2rad('10:22:14')
dms2rad('10d22m14s')
dms2rad('-0, 30')
dms2rad('-77d30.5m')
dms2rad('W 77d30.5m')
dms2rad(-77.5083333)
```

**Description**

Calculates source elevation and azimuth in degrees given declination, hour angle, and observatory latitude.

**Usage**

elev(dec.sou = "33d 09m 35.0s", ha = 0, lat.obs = "38d 25m 59.2s")

azimuth(dec.sou = "33d 09m 35.0s", ha = 0, lat.obs = "38d 25m 59.2s")
Arguments

dec.sou    Source declination (string)
ha         Hour angle (decimal hours)
lat.obs    Observatory latitude (string)

Details

Enter latitude as a text string with fields separated by characters d, h, m, s, a colon, or a comma, e.g. '38d25m59.2s' or '38, 25, 59.2' or '38:25:59.2' or '38:25.987' for the Green Bank Telescope. Spaces are removed in input conversion. Decimal values for degrees or minutes are allowed. Trailing missing values are taken as zero.

Value

Source elevation or azimuth (E from N) in degrees.

Note

Geometrical calculation only, no corrections for refraction, aberration, precession, etc.

Author(s)

Andrew Harris

References


See Also

dms2rad, hms2rad for input formats, ut2ha to convert UT to hour angle.

Examples

# Maximum elevation at Green Bank
elev(dms2rad('-28, 20'))

# Maximum elevation at Mauna Kea
elev(dms2rad('-28, 20'), 0, '19:49')

# Plot elevation and azimuth vs. hour angle
ha <- seq(0, 24, 0.25)
el <- elev('30d 33m 22s', ha)
plot(ha, el, t='l', col=4)
az <- azimuth('30d 33m 22s', ha)
plot(ha, az, t='l', col=4)

# Plot elevation and azimuth vs. UT (using many defaults)
h.ut <- seq(0, 24, 0.25)
el <- elev(dec.sou='30d 33m 22s', ha=ut2ha(hr=h.ut))
gmst1

GMST1 (Greenwich Mean Siderial Time at 0h, UT1) from UT1 date

Description

Calculate Greenwich Mean Siderial Time at 0h, UT1 (GMST1) from UT1 year, month, and day.

Usage

gmst1(yr = 2012, mo = 1, dy = 1)

Arguments

yr UT1 year (integer)
mo UT1 month (integer)
dy UT1 day (integer)

Details

Function calculates Greenwich Mean Siderial Time at 0h, UT1 (GMST1) given UT1 year, month, and day.

Value

Returns fractional hours of GMST1 with class fracHrs. The corresponding print method gives hh:mm:ss format rounded to n decimal places in seconds by setting options('digits.secs'=n).

Note

Multiply UT1 fractional day by 1.002737909350795 to get fractional sidereal day.

Author(s)

Andrew Harris

References

Explanatory Supplement to the Astronomical Almanac Seidelmann (ed), c. 1992

See Also

ymd2jd
Examples

```r
out <- gmst1(yr=2012, mo=7, dy=8)
str(out)
out
```

---

**hms2rad**  
*Hours, minutes, and seconds to radians*

### Description
Angular conversion from hours, minutes, and seconds to radians.

### Usage
```r
hms2rad(h = '12h 3m 45.6s')
```

### Arguments
- **h**: String hours, minutes, and seconds

### Details
Function reads a string (the input is a string to allow conversion of angles between -1 and zero hours) with hours, minutes, and seconds separated by any of characters d, m, s, a colon, or a comma. Spaces are not valid separators, as they are removed as part of input parsing. Zeros are the default if values for minutes or seconds are missing from the string. A minus sign before the hours indicates negative hours. Decimal values are allowed in any position.

### Value
Angle in radians.

### Author(s)
Andrew Harris

### See Also
dms2rad, rad2hms, rad2dms

### Examples
```r
hms2rad('10, 22, 14')
hms2rad('0:30')
hms2rad('0h30')
```
## Description
Precession from J1950 to B2000

## Usage
\[
\text{j2b}(ra = "17:30:30", \text{dec} = "-28:47")
\]

## Arguments
- **ra**: J2000 Right ascension (string)
- **dec**: J2000 Declination (string)

## Details
Enter positions as text strings with fields separated by characters d, h, m, s, a colon, or a comma, e.g. '17, 42, 28', '-28h43m03s', or '- 28 :43 : 3'. Spaces are removed in input conversion. Trailing missing values are taken as zero. The code uses an approximate formula for precession; spot checks give results accurate within a few tenths of an arcsecond.

## Value
List with strings in:
- **ra1950**: B1950 Right ascension
- **dec1950**: B1950 Declination

## Note
Values based on power-law expansion of more exact calculation.

## Author(s)
Andrew Harris

## References

## See Also
- **b2j**: See **dms2rad**, **hms2rad** for input conversions.
Examples

```r
j2b()
j2b(ra='17h43m', dec='-28d47m30s')
tmp <- j2b(ra='17, 43', dec='- 28, 47, 30')
str(tmp)
tmp
```

---

**jd2ymd**

*JD to year, month, date*

Description

Convert Julian date to UT1 year, month, and date.

Value

Date for 0h, UT1, with class POSIXct

Author(s)

Andrew Harris

References

Fliegel & Van Flandern, Comm. ACM 10, 657 (1968), whose algorithm uses FORTRAN integer mathematics

See Also

`weekdays, dmjd2ut`

Examples

```r
jd2ymd(2456092.5) # returns 0h date, 2012-06-14 UT
jd2ymd(2456092.6) # returns 0h date, 2012-06-14 UT
jd2ymd(2456092.4) # returns 0h date, 2012-06-13 UT
```
**planetFlux**

**Flux density from a thermal disk**

**Description**

The flux density from a disk-shaped blackbody with uniform temperature observed in a Gaussian beam.

**Usage**


```r
planetFlux(T = 195, dp = 14.8, thetab = 19.4, f = 32)
```

**Arguments**

- `T` : Disk’s physical temperature
- `dp` : Planet diameter, arcsec
- `thetab` : Beam FWHM, arcsec
- `f` : Observing frequency, GHz

**Details**

Geometry is for a uniform-temperature disk, a planet to some approximation, in a Gaussian beam.

**Value**

Flux density in janskys

**Note**

For a physical Mars model, see [http://www.aoc.nrao.edu/~bbutler/work/mars/model/](http://www.aoc.nrao.edu/~bbutler/work/mars/model/)

**Author(s)**

Andrew Harris

**Examples**

```r
planetFlux()
```
rad2dms

Convert radians to degrees, minutes, and seconds

Description
Angular conversion from radians to degrees, minutes, and seconds

Usage
rad2dms(rad = 1, places = 2)

Arguments
rad             Decimal radians
places          Number of decimal places in seconds term (0:6)

Details
Convert radians to degrees, minutes, and seconds.

Value
Fixed-format string with sign, then degrees, minutes, and seconds separated by colons.

Author(s)
Andrew Harris

See Also
rad2hms, dms2rad, hms2rad

Examples
rad2dms(2.44)
rad2dms(dms2rad(c('-1,4,5.12', '10:04: 5.3')), places=3)
rad2dms(-66.5 * pi/180) # from degrees to dms
rad2hms

Convert radians to hours, minutes, and seconds

Description
Angular conversion from radians to hours, minutes, and seconds

Usage
rad2hms(rad = 1, places = 1)

Arguments
- rad: Decimal radians
- places: Number of decimal places in seconds term (0:6)

Value
Fixed-format string with hours, minutes, and seconds separated by colons.

Author(s)
Andrew Harris

See Also
rad2dms, dms2rad, hms2rad

Examples
rad2hms(2.44)
rad2hms(hms2rad(c("10:04:5.12", "27,04,5.3", "-3:0:0")), places=3)
rad2hms(266.5 * pi/180) # from degrees to hms

ut2dmjd

UT to DMJD

Description
Universal time to decimal modified Julian date.

Usage
ut2dmjd(yr = 2012, mo = 1, dy = 1, hr = 0, mi = 0, se = 0)
ut2lst

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>yr</td>
<td>UT year</td>
</tr>
<tr>
<td>mo</td>
<td>UT month</td>
</tr>
<tr>
<td>dy</td>
<td>UT day</td>
</tr>
<tr>
<td>hr</td>
<td>UT hour</td>
</tr>
<tr>
<td>mi</td>
<td>UT minute</td>
</tr>
<tr>
<td>se</td>
<td>UT second</td>
</tr>
</tbody>
</table>

Value

Decimal modified Julian date.

Note

Uses `ymd2jd` to calculate Julian date

Author(s)

Andrew Harris

See Also

`dmjd2ut`

Examples

```r
ut2dmjd(yr=2000, mo=1, dy=1, hr=0, mi=0, se=0)
format(ut2dmjd(yr=2012, mo=5, dy=20, hr=7, mi=8, se=39), digits=10)
```

Description

Functions to calculate local sidereal time (LST) or hour angle (HA) from Universal time (strictly, UTC1).

Usage

```r
ut2lst(yr = 2012, mo = 1, dy = 1, hr = 0, mi = 0, se = 0,
  lon.obs = "W 79d 50.5m")
```

```r
ut2ha(yr = 2012, mo = 1, dy = 1, hr = 0, mi = 0, se = 0,
  ra.sou = "13h 31m 08.3s", lon.obs = "W 79d 50m 23.4s")
```
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>yr</td>
<td>UT1 Year</td>
</tr>
<tr>
<td>mo</td>
<td>UT1 Month number</td>
</tr>
<tr>
<td>dy</td>
<td>UT1 Day number</td>
</tr>
<tr>
<td>hr</td>
<td>UT1 Hour</td>
</tr>
<tr>
<td>mi</td>
<td>UT1 Minute</td>
</tr>
<tr>
<td>se</td>
<td>UT1 Seconds</td>
</tr>
<tr>
<td>ra.sou</td>
<td>String with source Right Ascension</td>
</tr>
<tr>
<td>lon.obs</td>
<td>String with observatory longitude</td>
</tr>
</tbody>
</table>

Details

If this input is `hr = Sys.time()` the function uses system time, including conversion to UT. UT is within a few seconds of UT1.

Value

Returns decimal local sidereal time in range 0 to 24 hours and hour angle from -1 to 12 hours, with class `fracHrs` (prints as h:m:s). For elapsed sidereal time difference over multiple sidereal days, difference UT days (from e.g. `ut2dmjd`) and multiply by 1.002737909350795.

Note

Spot checks show values match tabulated values in The Astronomical Almanac within ~0.01 seconds.

Author(s)

Andrew Harris

References


See Also

`ymd2jd`, `gmst1`, `dms2rad` and `hms2rad` for input formats, `Sys.time`, `Sys.timezone` and time zone examples in `as.POSIXlt`.

Examples

```r
# LST at UT1 midnight on the first of every month for Green Bank, WV, USA
midLST <- ut2lst(yr = 2012, mo = 1:12, dy = 1, hr = 0, mi = 0, se = 0, lon.obs="W 79d 50.5m")
str(midLST)
midLST
```
# LST at EST midnight on the first of every month for Green Bank, WV, USA
# (EST = UT1 - 5 hours)
midLST <- ut2lst(yr = 2012, mo = 1:12, dy = 1, hr = -5, mi = 0, se = 0,
lon.obs="W 79d 50.5m")
str(midLST)
midLST

# LST in Green Bank, WV, USA, now, and 12 hours from now.
ut2lst(Sys.time())
ut2lst(Sys.time() + 12*3600)

# Hour angle of 3C286 in Green Bank now (using function defaults)
ut2ha(Sys.time())

\begin{verbatim}

ymd2jd
\end{verbatim}

\textbf{ymd2jd}

\textit{Year, month, day to 0h on Julian day}

\textbf{Description}

Convert year, month, day to 0h on Julian day.

\textbf{Usage}

ymd2jd(yr = 2012, mo = 1, dy = 1)

\textbf{Arguments}

\begin{itemize}
  \item \textbf{yr} \hspace{1cm} \text{UT1 Year}
  \item \textbf{mo} \hspace{1cm} \text{UT1 Month number}
  \item \textbf{dy} \hspace{1cm} \text{UT1 Day number}
\end{itemize}

\textbf{Details}

Returns Julian date of 0 hours on the specified day. To get to noon on day, the time origin of Julian days, add 0.5.

\textbf{Value}

Julian date

\textbf{Author(s)}

Andrew Harris
References


See Also

weekdays, ut2dmjd

Examples

# Ensure enough digits to see result, then return to previous value
dig <-getOption('digits')
options(digits=16)
ymd2jd(yr=2000, mo=1, dy=1)
ymd2jd(yr=2000, mo=1, dy=1.3)  # rounds to nearest day
options(digits=dig)
jd2ymd(ymd2jd(yr=2000, mo=1, dy=1))
Index

* chron
  dmjd2ut, 5
gmst1, 9
jd2ymd, 12
ut2dmjd, 15
ut2lst, 16
ymd2jd, 18

* misc
  angSep, 2
b2j, 3
beamDiskOverlap, 4
dms2rad, 6
elev, 7
hms2rad, 10
j2b, 11
planetFlux, 13
rad2dms, 14
rad2hms, 15

* package
  astroFns-package, 2

angSep, 2
as.POSIXct, 6
as.POSIXlt, 17
astroFns (astroFns-package), 2
astroFns-package, 2
axis.POSIXct, 6
azimuth (elev), 7
b2j, 3, 11
beamDiskOverlap, 4
dmjd2ut, 5, 12, 16
dms2rad, 3, 4, 6, 8, 10, 11, 14, 15, 17
elev, 7
gmst1, 9, 17
hms2rad, 3, 4, 7, 8, 10, 11, 14, 15, 17

ISOdatetime, 6
j2b, 4, 11
jd2ymd, 12
planetFlux, 13
rad2dms, 7, 10, 14, 15
rad2hms, 7, 10, 14, 15
strptime, 6
Sys.time, 17
Sys.timezone, 6, 17
ut2dmjd, 6, 15, 17, 19
ut2ha, 8
ut2ha (ut2lst), 16
ut2lst, 16
weekdays, 12, 19
ymd2jd, 6, 9, 16, 17, 18