Package: galisats (via r-universe)

October 26, 2026

Title Configuration of Jupiter's Four Largest Satellites

Version 2.2.0

Description Calculate, plot and animate the configuration of Jupiter's four largest satellites (known as Galilean satellites) for a given date and time (UTC - Coordinated Universal Time). The galsat() function returns numerical values of the satellites' positions. x – the apparent rectangular coordinate of the satellite with respect to the center of Jupiter's disk in the equatorial plane in the units of Jupiter's equatorial radius; X is positive toward the west, y – the apparent rectangular coordinate of the satellite with respect to the center of Jupiter's disk from the equatorial plane in the units of Jupiter's equatorial radius; Y is positive toward the north. For more details see Meeus (1988, ISBN 0-943396-22-0) ``Astronomical Formulae for Calculators''. The galsat_animate() function creates an animation of the Galilean satellites' positions. You provide the starting time, duration, the time step between frames, and the pause between frames. The function delta_t() returns the value of delta-T in units of seconds.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports graphics, png

URL https://lechjaszowski.github.io/galilean_satellites/

Config/pak/sysreqs libpng-dev

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

Date/Publication 2025-09-27 07:35:20 UTC

RemoteUrl https://github.com/lechjaszowski/galilean_satellites

RemoteRef HEAD

RemoteSha f6251e2fb4a91bc55a0182061413eb26719b2757

Contents

delta_t 2
galsat 3
galsat_animate 4

Index **6**

`delta_t` *Return the value of delta-T in units of seconds*

Description

Converting the Coordinated Universal Time (UTC) to the Ephemeris Time (ET) is complex. It is due to the unpredictable nature of the Earth's rotation, which is the basis for UTC, whereas ET was based on the more uniform orbital motion of the Earth around the Sun. The key to converting between these time scales lies in a value known as delta-T, which is the difference between a uniform time scale and one based on Earth's rotation. The conversion is handled as: ET = UTC + deltaT However, delta-T is not a constant value and cannot be calculated using a simple formula. The delta-T values are derived from the historical records and from direct observations. A series of polynomial expressions have been created to simplify the evaluation of delta-T. The calculated values are valid for the years from -1999 to +3000.

Usage

```
delta_t(year, month)
```

Arguments

`year`	Type in the year (integer between -1999 and 3000).
`month`	Type in the month (integer between 1 and 12).

Details

More details: Morrison, L. and Stephenson, F. R., "Historical Values of the Earth's Clock Error delta-T and the Calculation of Eclipses", J. Hist. Astron., Vol. 35 Part 3, August 2004, No. 120, pp 327-336 (2004) Stephenson F.R., Historical Eclipses and Earth's Rotation, Cambridge Univ. Press, 1997

Value

`numeric`: vector of numeric values

Examples

```
delta_t(1999, 10)
delta_t(c(-200, 1610, 2030), c(1, 10, 12))
```

`galsat` *Calculate & draw the positions of the Galilean satellites*

Description

`galsat()` is used to determine the positions of the four greatest satellites of Jupiter (called Galilean satellites). Positions are shown on the plot for given UTC time (Coordinated Universal Time between year 0 and 3000) with respect to the planet, as seen from the Earth.

The `galsat()` function returns numerical values of the satellites' positions:

x - the apparent rectangular coordinate of the satellite with respect to the center of Jupiter's disk in the equatorial plane in the units of Jupiter's equatorial radius; X is positive toward the west

y - the apparent rectangular coordinate of the satellite with respect to the center of Jupiter's disk from the equatorial plane in the units of Jupiter's equatorial radius; Y is positive toward the north

u_corrected - the corrected angular position of the satellite in degrees, used to determine visibility conditions (whether a moon can be seen against Jupiter's disk or is hidden behind it)

Usage

```
galsat(year, month, day, hour, minute)
```

Arguments

`year`	Type in the year (integer number from 0 to 3000).
`month`	Type in the month (integer number from 1 to 12).
`day`	Type in the day (integer number from 1 to 31).
`hour`	Type in the hour (integer number from 0 to 23).
`minute`	Type in the minute (integer number from 0 to 59).

Details

The function is based on algorithms in the book: Astronomical Formulae for Calculators (4th edition), Jean Meeus, Willmann-Bell Inc., 1988

Value

`data.frame`: 4 observations of 4 variables: $ moon : chr "Io" "Europa" "Ganymede" "Callisto" $ x : num $ y : num $ u_corrected: num Four rows - each row has the position (x,y) and corrected angular position (u_corrected) of one moon. Additionally, the positions of the moons are shown graphically.

Examples

```
galsat(2025, 10, 13, 23, 30)
# Also try these interesting configuration moments:
galsat(2021, 8, 15, 15, 48)
galsat(2032, 1, 5, 6, 44)
galsat(2033, 7, 28, 4, 50)
galsat(2039, 7, 31, 18, 55)
```

---

galsat_animate *Animate the motion of Jupiter's Galilean satellites*

Description

`galsat_animate()` creates an animation showing the orbital motion of Jupiter's four largest satellites over time. The function starts from a user-specified time and advances in regular intervals to show how the moons move around Jupiter.

Usage

```
galsat_animate(
  year,
  month,
  day,
  hour,
  minute,
  duration_hours = 24,
  time_step_minutes = 5,
  pause_seconds = 0.05
)
```

Arguments

`year`	Type in the starting year (integer number from 0 to 3000).
`month`	Type in the starting month (integer number from 1 to 12).
`day`	Type in the starting day (integer number from 1 to 31).
`hour`	Type in the starting hour (integer number from 0 to 23).
`minute`	Type in the starting minute (integer number from 0 to 59).
`duration_hours`	Duration of the animation in hours (default 24).
`time_step_minutes`	Time step between frames in minutes (default 5).
`pause_seconds`	Pause between frames in seconds (default 0.05).

Details

The animation uses the galsat() function internally to calculate satellite positions at each time step. Time is advanced by the specified interval (default 5 minutes) for each frame of the animation.

Value

Creates an animated plot showing the orbital motion of Jupiter's moons. Returns invisibly the final positions data frame.

Examples

```
# Animate 2 hours with 10-minutes steps
galsat_animate(2025, 10, 6, 21, 50, duration_hours = 2, time_step_minutes = 10)
```

Index

delta_t, 2

galsat, 3
galsat_animate, 4