Package: PCICt (via r-universe)

November 13, 2026

Version 0.5.4.4.9001

Title Implementation of POSIXct Work-Alike for 365 and 360 Day Calendars

Author David Bronaugh <bronaugh@uvic.ca> for the Pacific Climate Impacts Consortium (PCIC); portions based on code written by the R-Core team and Ulrich Drepper.

Maintainer James Hiebert <hiebert@uvic.ca>

Depends R (>= 2.12.0), methods, graphics

Suggests RUnit

Description Provides a work-alike to R's POSIXct class which implements 360- and 365-day calendars in addition to the gregorian calendar. Compatibility updates by Renova.

License GPL-2

URL https://www.r-project.org

NeedsCompilation yes

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

Date/Publication 2026-11-13 17:54:25 UTC

RemoteUrl https://github.com/renova-r/PCICt

RemoteRef renova-0.5.4.4.9001

RemoteSha 73343482b78ad44c02f8e6eace0b4f0ea8767075

Contents

as.PCICt 2
c.PCICt 4
Ops.PCICt 4
PCICt 5
round.PCICt 6

Index **7**

as.PCICt *PCICt*

Description

These functions convert between PCICt objects and other types of data.

Usage

```
## S3 method for class 'PCICt'
as(x, cal, ...)
.PCICt(x, cal)
## S3 method for class 'numeric'
as.PCICt(x, cal, origin, ...)
## Default S3 method:
as.PCICt(x, cal, format, ...)
## S3 method for class 'POSIXlt'
as.PCICt(x, cal, ...)
## S3 method for class 'POSIXct'
as.PCICt(x, cal, ...)
## S3 method for class 'PCICt'
as.POSIXct(x, tz="", ...)
## S3 method for class 'PCICt'
as.POSIXlt(x, tz="", ...)
## S3 method for class 'PCICt'
as.character(x, ...)
```

Arguments

x	the input data.
cal	the calendar type.
origin	the origin for a numeric time.
tz	the time zone to put the data in.
format	the format to parse the date using.
...	any additional arguments passed on.

Details

as.PCICt converts the x argument, where x is of type POSIXlt, POSIXct, or character, to a PCICt object with the given calendar type. Calendar types include 360 day calendars("360_day", "360"), 365 day calendars ("365_day", "365", "noleap"), and Gregorian calendars ("gregorian", "proleptic_gregorian"). When converting a character object, one can also specify the format with fmt=, which uses a format documented in the help page for strptime.

.PCICt converts numeric objects into PCICt objects, using x as seconds since 1970-01-01 and applying the supplied calendar to the data.

as.PCICt.numeric converts numeric objects into PCICt objects, using `x` as seconds since the origin. `origin` can be either a character or a PCICt object.

as.POSIXct.PCICt and as.POSIXlt.PCICt convert PCICt objects into POSIXct or POSIXlt objects, respectively. With POSIXct objects, this may result in apparent gaps in the timeseries, and the transformation will not be trivially reversible. See the example below for how to transition between PCICt and POSIXct.

as.character.PCICt converts a PCICt object to a character string representation of that object.

as.PCICt.default, as.PCICt.POSIXct, and as.PCICt.POSIXlt are helpers which are called by as.PCICt. Normally you will not need to call them directly.

Value

For as.PCICt and .PCICt, a PCICt object with the given calendar type.

For as.POSIXct.PCICt and as.POSIXlt.PCICt, a POSIXct or POSIXlt object, respectively.

See Also

`as.POSIXlt`, `as.POSIXct`, `strptime`

Examples

```
## Convert these strings to PCICt objects.
x <- as.PCICt(c("1961-09-02", "1963-02-01"), cal="360_day")

## Convert these strings to POSIXlt objects, then coerce them into PCICt objects.
y <- as.POSIXlt(c("1961-09-02", "1963-02-01"))
x <- as.PCICt(y, cal="360_day")

## Convert a string to PCICt using a format string.
q <- as.PCICt("03292001", cal="365_day", format="%m%d%Y")

## This will cause a parsing error.
## Not run: bad.r <- as.PCICt("moo", cal="365_day")

## Convert a POSIXct to PCICt 360-day
foo <- as.POSIXct("2011-04-01")
bar <- as.PCICt(as.character(foo), cal="360_day")

## Test whether the result is the same
baz <- as.PCICt("2011-04-01", cal="360_day")
bar == baz

## Convert a sequence of days plus an origin to PCICt (as seen in NetCDF files)
cal <- "365_day"
origin <- "1968-01-01"
seconds.per.day <- 86400
ts.dat.days <- 0:100
origin.pcict <- as.PCICt(origin, cal)
ts.dat.pcict <- origin.pcict + (ts.dat.days * seconds.per.day)
```

c.PCICt *c.PCICt*

Description

These functions implement repetition, cutting, sequences, truncation, means, ranges, min, max, julian days, differences, concatenation, and tests for numericity.

See Also

c, rep, seq, trunc, mean, range, julian, diff, cut, is.numeric, max, min.

Ops.PCICt *Ops.PCICt*

Description

These functions implement subtraction, addition, indexing, and index assignment operations as in POSIXct.

Value

A PCICt object with the given operations performed.

Examples

```
## Create a list of PCICt of length 2 with a 365-day calendar
x <- as.PCICt(c("1961-09-02", "1963-02-01"), cal="365_day")

## Look at the difference between the two elements of x
y <- x[1] - x[2]

## Change the first element of x
x[1] <- as.PCICt("1962-09-02", cal="365_day")
```

PCICt *PCICt, a POSIXct work-alike for 360- and 365-day calendars*

Description

This package implements a work-alike to R's POSIXct class which implements 360- and 365-day calendars in addition to the gregorian calendar.

Details

Many global climate models (GCMs) and regional climate models (RCMs) are run using an idealized and simplified calendar which only includes 365 days or 360 days per year. When trying to do seasonal or monthly analysis on a set of models which use different calendar types, analyses may not be comparable unless one can normalize the calendars and the times which are represented therin. Thing get even more difficult when trying to compare model output with observations data which is located in a particular time on the Gregorian calendar.

The PCICt package attempts to solve this problem by creating a new time type, PCICt, which inherits from the POSIXt type. All of the functionality provided by POSIXt is also provided by PCICt, however PCICt does the work of normalizing the calendars and making points in time on seperate calendars cross comparable.

365-day calendars are implemented using a 365-day non-leap year from a Gregorian calendar.

360-day calendars are not implemented as 12 equal months of 30 days. They are implemented as 12 months of the following lengths, in days, with the first month being January: 30, 28, 31, 30, 30, 30, 30, 31, 30, 30, 30, 30. This was a decision to ease implementation.

To map a 365_day calendar to Gregorian, PCICt simply drops February 29 from leap years. To map a 360_day calendar to Gregorian, PCICt attempts to remap the days such that the five lost days are distributed as equally as possible across the seasons (winter loses two days while, spring/summer/fall each lose one).

There are a few problems with this implementation. As noted above, 360-day calendars do not use equal months, which may cause problems in certain situations.

Another problem originates from within R itself. Many functions in R strip attributes from data. If this happens to a PCICt object, it cannot be coerced back into a PCICt, as it is lacking the calendar attribute. This causes problems with several internal functions, like mean. PCICt includes a wrapper for mean. You will probably run into these problems. When you do, please use the wrapper for mean as a template for your wrapper.

This package may be modified substantially in future to solve these problems.

References

http://www.pacificclimate.org

See Also

as.PCICt

round.PCICt *round.PCICt*

Description

Round PCICt objects to the nearest second/minute/hour/day

Usage

```
## S3 method for class 'PCICt'
round(x, digits = c("secs", "mins", "hours", "days"))
```

Arguments

x	Dates to be rounded.
digits	Unit to round the dates to.

Details

round.PCICt rounds the dates in the x argument to the nearest second/minute/hour/day, as specified by the poorly named digits argument.

Value

The dates in x, rounded to the nearest second/minute/hour/day.

See Also

trunc

Examples

```
## Convert strings to PCICt objects, on a 360 day calendar
x <- as.PCICt(c("1961-02-30 12:00:00", "1962-03-24 12:34:56"), cal="360")

## Round them to the nearest hour
x.hour <- round(x, "hours")

## Round them to the nearest day
x.day <- round(x, "days")
```

Index

∗ **calendar**
    PCICt, 5
∗ **chron**
    c.PCICt, 4
    PCICt, 5
∗ **climate**
    PCICt, 5
∗ **date**
    PCICt, 5
∗ **time**
    PCICt, 5
∗ **ts**
    as.PCICt, 2
    c.PCICt, 4
    Ops.PCICt, 4
    PCICt, 5
∗ **utilities**
    PCICt, 5
+.PCICt *(*Ops.PCICt*)*, 4
-.PCICt *(*Ops.PCICt*)*, 4
.PCICt *(*as.PCICt*)*, 2
[.PCICt *(*Ops.PCICt*)*, 4
[<-.PCICt *(*Ops.PCICt*)*, 4

as.character.PCICt *(*as.PCICt*)*, 2
as.PCICt, 2, *5*
as.POSIXct, *3*
as.POSIXct.PCICt *(*as.PCICt*)*, 2
as.POSIXlt, *3*
as.POSIXlt.PCICt *(*as.PCICt*)*, 2

c, *4*
c.PCICt, 4
cut, *4*
cut.PCICt *(*c.PCICt*)*, 4

diff, *4*
diff.PCICt *(*c.PCICt*)*, 4

is.numeric, *4*
is.numeric.PCICt *(*c.PCICt*)*, 4

julian, *4*
julian.PCICt *(*c.PCICt*)*, 4

max, *4*
max.PCICt *(*c.PCICt*)*, 4
mean, *4*
mean.PCICt *(*c.PCICt*)*, 4
min, *4*
min.PCICt *(*c.PCICt*)*, 4

Ops.PCICt, 4

PCICt, 5
PCICt-package *(*PCICt*)*, 5

range, *4*
range.PCICt *(*c.PCICt*)*, 4
rep, *4*
rep.PCICt *(*c.PCICt*)*, 4
round.PCICt, 6

seq, *4*
seq.PCICt *(*c.PCICt*)*, 4
strptime, *3*

trunc, *4*, *6*
trunc.PCICt *(*c.PCICt*)*, 4