Package ‘aion’

Title Archaeological Time Series

Version 1.0.4

Maintainer Nicolas Frerebeau <nicolas.frerebeau@u-bordeaux-montaigne.fr>

Description A toolkit for archaeological time series. This package provides a system of classes and methods to represent and work with archaeological time series. Dates are represented as "rata die" and can be converted to (virtually) any calendar defined by Reingold and Dershowitz (2018) <doi:10.1017/9781107415058>. This packages offers a simple API that can be used by other specialized packages.

License GPL (>= 3)


BugReports https://github.com/tesselle/aion/issues

Depends R (>= 3.3)

Imports arkhe (>= 1.6.0), graphics, grDevices, methods

Suggests folio (>= 1.4.0), knitr, markdown, rsvg, svglite, tinysnapshot, tinytest

VignetteBuilder knitr

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Collate 'AllClasses.R' 'AllGenerics.R' 'aion-internal.R'
'aion-package.R' 'calendar-gregorian.R' 'calendar-julian.R'
'calendar.R' 'coerce.R' 'convert.R' 'data.R' 'mutators.R'
'operators.R' 'plot.R' 'series.R' 'show.R' 'subset.R' 'time.R'
'validate.R' 'years.R' 'zzz.R'

NeedsCompilation no

Author Nicolas Frerebeau [aut, cre] (<https://orcid.org/0000-0001-5759-4944>, Université Bordeaux Montaigne), Joe Roe [aut] (<https://orcid.org/0000-0002-1011-1244>, Universität
**Contents**

- arithmetic .......................................................... 3
- as_date .............................................................. 4
- as_decimal ........................................................... 5
- as_fixed .............................................................. 6
- as_year ............................................................... 7
- calendar ............................................................. 9
- calendar_get ........................................................ 10
- convert ............................................................... 12
- data.frame .......................................................... 13
- dates ................................................................. 14
- fixed ................................................................. 15
- fixed_gregorian ..................................................... 16
- fixed_julian ......................................................... 18
- format ............................................................... 19
- gregorian ............................................................ 20
- image ................................................................. 21
- is ..................................................................... 22
- julian ................................................................. 23
- plot ................................................................. 24
- pretty ............................................................... 26
- RataDie-class ....................................................... 27
- series ............................................................... 27
- span ................................................................. 29
- start ................................................................. 31
- subset ............................................................... 32
- time ................................................................. 33
- TimeScale-class .................................................... 34
- TimeSeries-class .................................................. 35
- window ............................................................... 35
- year_axis ............................................................ 36

**Index** 38
Description

Operators performing arithmetic operations.

Usage

```r
## S4 method for signature 'RataDie,RataDie'
Arith(e1, e2)
```

```r
## S4 method for signature 'numeric,RataDie'
Arith(e1, e2)
```

```r
## S4 method for signature 'RataDie,numeric'
Arith(e1, e2)
```

Arguments

- `e1`, `e2` A `RataDie` object or a `numeric` vector.

Details

`RataDie` will be converted to a plain numeric vector if a computation no longer makes sense in temporal terms.

Value

A logical vector.

Author(s)

N. Frerebeau

See Also

Other fixed date tools: `as_date()`, `as_decimal()`, `as_fixed()`, `as_year()`, `fixed()`, `fixed_gregorian`, `fixed_julian`, `format()`, `pretty()`

Examples

```r
## Vectors of years
x <- fixed(c(-350, 31, 1072, 576, 1130), calendar = CE())
y <- fixed(c(1494, 1645, -869, 1440, 1851), calendar = CE())

## Move forward in time
x + y
```
## Move backward in time
\[ x - y \]

## Not rata die anymore
\[ x \times y \]

---

**as_date**  
*Date Conversion from Rata Die*

**Description**  
Date Conversion from *Rata Die*

**Usage**

```r
as_date(object, calendar)
```

### S4 method for signature 'numeric,GregorianCalendar'

```r
as_date(object, calendar)
```

### S4 method for signature 'numeric,JulianCalendar'

```r
as_date(object, calendar)
```

**Arguments**

- `object` A *RataDie* object (see `fixed()`).
- `calendar` A *TimeScale* object specifying the target calendar (see `calendar()`).

**Value**

A *numeric* vector of (decimal) years.

**Author(s)**

N. Frerebeau

**References**


**See Also**

Other fixed date tools: `arithmetic`, `as_decimal()`, `as_fixed()`, `as_year()`, `fixed()`, `fixed_gregorian`, `fixed_julian`, `format()`, `pretty()`
Examples

```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)
```

---

**as_decimal**

*Converts a Date to a Decimal of its Year*

**Description**

Converts a Date to a Decimal of its Year

**Usage**

```r
as_decimal(year, month, day, calendar)
```

**Arguments**

- `year`: A numeric vector of years. If month and day are missing, decimal years are expected.
- `month`: A numeric vector of months.
- `day`: A numeric vector of days.
- `calendar`: A TimeScale object specifying the calendar of year, month and day (see `calendar()`).

**Value**

A numeric vector of decimal years.

**Author(s)**

N. Frerebeau
See Also

Other fixed date tools: arithmetic, as_date, as_fixed, as_year, fixed, fixed_gregorian, fixed_julian, format, pretty

Examples

```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)
```

---

as_fixed

Coerce to Rata Die

Description

Coerce to Rata Die

Usage

```r
as_fixed(from)
```

## S4 method for signature 'numeric'

```r
as_fixed(from)
```

Arguments

```r
from          A numeric vector of rata die.
```

Value

```r
A RataDie object.
```

Author(s)

N. Frerebeau
References

See Also
Other fixed date tools: arithmetic, as_date(), as_decimal(), as_year(), fixed(), fixed_gregorian, fixed_julian, format(), pretty()

Examples
```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)
```

---

as_year | Year Conversion from Rata Die

### Description
Year Conversion from *Rata Die*

### Usage
```r
as_year(object, calendar, ...)
```

**S4 method for signature 'numeric,GregorianCalendar'**
```r
as_year(object, calendar, decimal = TRUE)
```

**S4 method for signature 'numeric,JulianCalendar'**
```r
as_year(object, calendar)
```
Arguments

- **object**: A `RataDie` object (see `fixed()`).
- **calendar**: A `TimeScale` object specifying the target calendar (see `calendar()`).
- ... Currently not used.
- **decimal**: A `logical` scalar: should decimal years be returned? If `FALSE`, the decimal part is dropped.

Value

A numeric vector of (decimal) years.

Author(s)

N. Frerebeau

References


See Also

Other fixed date tools: `arithmetic, as_date(), as_decimal(), as_fixed(), fixed(), fixed_gregorian, fixed_julian, format(), pretty()`

Examples

```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)
```
Description

Calendar

Usage

calendar(object)

## S4 method for signature 'character'
calendar(object)

Arguments

object A character string specifying the abbreviated label of the time scale (see details) or an object from which to extract the time scale.

Details

The following time scales are available:

<table>
<thead>
<tr>
<th>label</th>
<th>era</th>
<th>calendar</th>
</tr>
</thead>
<tbody>
<tr>
<td>BP</td>
<td>Before Present</td>
<td>Gregorian</td>
</tr>
<tr>
<td>BC</td>
<td>Before Christ</td>
<td>Gregorian</td>
</tr>
<tr>
<td>BCE</td>
<td>Before Common Era</td>
<td>Gregorian</td>
</tr>
<tr>
<td>AD</td>
<td>Anno Domini</td>
<td>Gregorian</td>
</tr>
<tr>
<td>CE</td>
<td>Common Era</td>
<td>Gregorian</td>
</tr>
<tr>
<td>b2k</td>
<td>Years before 2000</td>
<td>Gregorian</td>
</tr>
<tr>
<td>julian</td>
<td></td>
<td>Julian</td>
</tr>
</tbody>
</table>

Value

A TimeScale object.

Note

Inspired by era::era() by Joe Roe.

Author(s)

N. Frerebeau

See Also

Other calendar tools: calendar_get, convert(), gregorian.is(), julian()
Examples

```r
## Define time scales
calendar("BP")
calendar("AD")
calendar("julian")

## Shortcuts
BP()
AD()
J()
```

<table>
<thead>
<tr>
<th>calendar_get</th>
<th>Calendar Parameters</th>
</tr>
</thead>
</table>

Description

Calendar Parameters

Usage

```r
calendar_label(object)
calendar_name(object)
calendar_unit(object)
calendar_epoch(object)
calendar_fixed(object)
calendar_direction(object)
```

```r
## S4 method for signature 'TimeScale'
calendar_label(object)
```

```r
## S4 method for signature 'TimeScale'
calendar_name(object)
```

```r
## S4 method for signature 'TimeScale'
calendar_unit(object)
```

```r
## S4 method for signature 'TimeScale'
calendar_epoch(object)
```

```r
## S4 method for signature 'TimeScale'
calendar_fixed(object)
```
calendar_direction(object)

Arguments

object A TimeScale object.

Value

- calendar_label() returns a character string specifying the abbreviated label of the time scale.
- calendar_name() returns a character string specifying the name of the time scale.
- calendar_unit() returns a character string specifying the name of the calendar.
- calendar_fixed() returns a length-one numeric vector specifying the reference date of the calendar (in rata die).
- calendar_epoch() returns a length-one numeric vector specifying the epoch year from which years are counted (starting date of the calendar, in years).
- calendar_direction() returns a length-one integer vector specifying if years are counted backwards (−1) or forwards (1) from epoch. Only the sign of calendar_direction() is relevant.

Author(s)

N. Frerebeau

See Also

Other calendar tools: calendar(), convert(), gregorian, is(), julian()

Examples

## Define time scales
calendar("BP")
calendar("AD")
calendar("julian")

## Shortcuts
BP()
AD()
J()
Description
Interconverts dates in a variety of calendars.

Usage
convert(from, to, ...)

## S4 method for signature 'character,character'
convert(from, to)

## S4 method for signature 'TimeScale,TimeScale'
convert(from, to)

Arguments
from A TimeScale object describing the source calendar.
to A TimeScale object describing the target calendar.
... Currently not used.

Value
A function that when called with a single numeric argument (fractional years) converts years from one calendar to another.

Author(s)
N. Frerebeau

See Also
Other calendar tools: calendar(), calendar_get, gregorian, is(), julian()

Examples
## Define time scales
BP <- calendar("BP")
AD <- calendar("AD")

## Make conversion functions
BP_to_AD <- convert(BP, AD)
AD_to_BP <- convert(AD, BP)

## Convert years
BP_to_AD(0)
AD_to_BP(1950)
Coerce to a Data Frame

Description
Coerce to a Data Frame

Usage
```r
## S4 method for signature 'TimeSeries'
as.data.frame(x, ..., calendar = NULL)
```

Arguments
- `x` A `TimeSeries` object.
- `...` Further parameters to be passed to `data.frame()`.
- `calendar` A `TimeScale` object specifying the target calendar (see `calendar()`). If `NULL` (the default), `rata die` are returned.

Value
A long `data.frame` with the following columns:
- `time` The (decimal) years at which the time series was sampled.
- `series` The name of the time series.
- `variable` The name of the variables.
- `value` The observed value.

Author(s)
N. Frerebeau

See Also
Other time series tools: `series()`, `span()`, `start()`, `time()`, `window()`

Examples
```r
## Create time-series of 20 observations
## Univariate
## Sampled every years starting from 1029 BCE
(X <- series(rnorm(30), time = 1029:1000, calendar = BCE()))

## Terminal and sampling times (returns rata die)
start(X)
end(X)
time(X)
```
## Multivariate
## Sampled every century starting from 1000 CE
(Y <- series(matrix(rnorm(90), 30, 3), time = 1000:1029, calendar = CE()))

## Terminal and sampling times (returns Gregorian Common Era years)
start(Y, calendar = CE())
end(Y, calendar = CE())
time(Y, calendar = CE())
span(Y, calendar = CE())

## Coerce to data frame
df <- as.data.frame(Y, calendar = BP())
head(df)

---

***Sample Data from Reingold and Dershowitz (2018)***

### Description
A dataset of 33 dates from the years -1000 to 2100 with their equivalents on different calendars.

### Usage
dates

### Format
A *data.frame* with 33 rows and 14 variables:

- **rata_die** Rata die.
- **weekday** Week day.
- **jd** Julian day.
- **mjd** Modified Julian day.
- **unix** Unix.
- **gregorian_year**, **gregorian_month**, **gregorian_day** Gregorian date.
- **julian_year**, **julian_month**, **julian_day** Julian date.
- **egyptian_year**, **egyptian_month**, **egyptian_day** Egyptian date.

### References
Description

*Rata Die* (Fixed Date)

Usage

```r
class(fixed) <- setGeneric("fixed")
setMethods(f"fixed", signature="numeric,missing,missing,GregorianCalendar" = function(year, calendar, scale = 1) {
  fixed(year, calendar, scale = 1)
})
```

Arguments

- **year**: A *numeric* vector of years. If `month` and `day` are missing, decimal years are expected.
- **month**: A *numeric* vector of months.
- **day**: A *numeric* vector of days.
- **calendar**: A *TimeScale* object specifying the calendar of year, month and day (see `calendar()`).
- **scale**: A length-one *integer* vector specifying the number of years represented by one unit. It should be a power of 10 (i.e. 1000 means ka).

Details

*Rata die* are represented as the number of days since 01-01-01 (Gregorian), with negative values for earlier dates.

Value

A *RataDie* object.

Author(s)

N. Frerebeau
References

See Also
Other fixed date tools: `arithmetic`, `as_date()`, `as_decimal()`, `as_fixed()`, `as_year()`, `fixed_gregorian`, `fixed_julian`, `format()`, `pretty()`

Examples
```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)
```

fixed_gregorian  Rata Die  Conversion to and from Gregorian Years

Description
Convenient functions for conversion from and to *rata die* for a given Gregorian era.

Usage
```r
fixed_from_BP(year, month, day)
fixed_to_BP(object)
fixed_from_BC(year, month, day)
fixed_to_BC(object)
fixed_from_BCE(year, month, day)
fixed_to_BCE(object)
```
fixed_gregorian

fixed_from_AD(year, month, day)

fixed_to_AD(object)

fixed_from_CE(year, month, day)

fixed_to_CE(object)

fixed_from_b2k(year, month, day)

fixed_to_b2k(object)

Arguments

year A numeric vector of years. If month and day are missing, decimal years are expected.
month A numeric vector of months.
day A numeric vector of days.
object A RataDie object (see fixed()).

Details

The astronomical notation is used for Gregorian years (there is a year 0).

Value

• fixed_from_*() returns a RataDie object.
• fixed_to_*() returns a numeric vector of Gregorian years.

Author(s)

N. Frerebeau

References


See Also

Other fixed date tools: arithmetic, as_date(), as_decimal(), as_fixed(), as_year(), fixed(), fixed_julian, format(), pretty()

Examples

```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
```
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)

---

**fixed_julian**

Rata Die Conversion to and from Julian Years

**Description**

Convenient functions for conversion from and to *rata die*.

**Usage**

```r
fixed_from_julian(year, month, day)

fixed_to_julian(object)
```

**Arguments**

- **year**: A numeric vector of years. If month and day are missing, decimal years are expected.
- **month**: A numeric vector of months.
- **day**: A numeric vector of days.
- **object**: A RataDie object (see `fixed()`).

**Value**

- `fixed_from_julian()` returns a RataDie object.
- `fixed_to_julian()` returns a numeric vector of Julian years.

**Author(s)**

N. Frerebeau

**References**

See Also

Other fixed date tools: arithmetic.as_date(), as_decimal(), as_fixed(), as_year(), fixed(), fixed_gregorian, format(), pretty()

Examples

```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)
```

Date Conversion to Character

Description

Date Conversion to Character

Usage

```r
## S4 method for signature 'TimeScale'
format(x)

## S4 method for signature 'RataDie'
format(
  x,
  prefix = c("a", "ka", "Ma", "Ga"),
  label = TRUE,
  calendar = getOption("aion.calendar")
)
```

Arguments

- `x` A `RataDie` object.
- `prefix` A `character` string specifying the prefix. It should be one of "a", "ka", "Ma" or "Ga". If `TRUE`, a good guess for an appropriate format is made.
label  A logical scalar: should the label of the calendar be displayed?
calendar  A TimeScale object specifying the target calendar (see calendar()).

Value
A character vector representing the date.

Author(s)
N. Frerebeau

See Also
Other fixed date tools: arithmetic, as_date(), as_decimal(), as_fixed(), as_year(), fixed(), fixed_gregorian, fixed_julian, pretty()

Examples
```r
## R 1.0.0
(y <- fixed(year = 2000, month = 02, day = 29, calendar = CE()))
as_date(y, calendar = CE())
as_year(y, calendar = CE())

## Create a vector of years BP (Gregorian)
## (every two years starting from 2000 BP)
(years <- seq(from = 2000, by = -2, length.out = 10))
## Convert years to rata die
(rd <- fixed(years, calendar = BP()))
## Convert back to Gregorian years BP
as_year(rd, calendar = BP())

## More convenient
(rd <- fixed_from_BP(years))
fixed_to_BP(rd)
```
BCE(...)  
AD(...)  
CE(...)  

Arguments  
...  
Currently not used.

Value  
A `GregorianCalendar` object.

Author(s)  
N. Frerebeau

See Also  
calendar()  
Other calendar tools: calendar(), calendar_get, convert(), is(), julian()

Examples  
## Define time scales  
calendar("BP")  
calendar("AD")  
calendar("julian")  

## Shortcuts  
BP()  
AD()  
J()  

---  

**image**  

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Heat Map</td>
</tr>
</tbody>
</table>

**Usage**  

## S4 method for signature 'TimeSeries'  
image(x, calendar = getOption("aion.calendar"), k = 1, ...)
Arguments

  x      A TimeSeries object.
calendar A TimeScale object specifying the target calendar (see calendar()).
  k      An integer specifying the slice of x along the third dimension to be plotted.
  ...   Further parameters to be passed to graphics::image().

Value

  image() is called for its side-effects: it results in a graphic being displayed. Invisibly returns x.

Author(s)

  N. Frerebeau

See Also

  graphics::image()
  Other plotting tools: plot(), year_axis()

Examples

  ## Create 6 time-series of 50 observations
  ## Sampled every two years starting from 2000 BP
  X <- series(
    object = matrix(rnorm(300), nrow = 50, ncol = 6),
    time = seq(2000, by = -2, length.out = 50),
    calendar = BP()
  )

  ## Image
  image(X, calendar = CE())

is

Is an Object a Calendar?

Description

  Test inheritance relationships between an object and a calendar class.

Usage

  is_gregorian(object)
  is_julian(object)

  ## S4 method for signature 'ANY'
  is_gregorian(object)
## S4 method for signature 'ANY'
is_julian(object)

**Arguments**

object Any R object.

**Value**

A logical vector.

**Author(s)**

N. Frerebeau

**See Also**

Other calendar tools: `calendar()`, `calendar.get.convert()`, `gregorian`, `julian()`

---

### julian

**Julian Calendar**

**Description**

Julian Calendar

**Usage**

`J(...)`

**Arguments**

... Currently not used.

**Value**

A `JulianCalendar` object.

**Author(s)**

N. Frerebeau

**See Also**

`calendar()`

Other calendar tools: `calendar()`, `calendar.get.convert()`, `gregorian`, `is()`
Examples

```
## Define time scales
calendar("BP")
calendar("AD")
calendar("julian")

## Shortcuts
BP()
AD()
J()
```

Description

Plot Time Series

Usage

```
## S4 method for signature 'TimeSeries,missing'
plot(
  x,
  facet = c("multiple", "single"),
  calendar = getOption("aion.calendar"),
  panel = graphics::lines,
  flip = FALSE,
  ncol = NULL,
  xlab = NULL,
  ylab = NULL,
  main = NULL,
  sub = NULL,
  ann = graphics::par("ann"),
  axes = TRUE,
  frame.plot = axes,
  panel.first = NULL,
  panel.last = NULL,
  ...
)
```

Arguments

- **x** A `TimeSeries` object.
- **facet** A character string specifying whether the series should be plotted separately (with a common time axis) or on a single plot? It must be one of "multiple" or "single". Any unambiguous substring can be given.
- **calendar** A `TimeScale` object specifying the target calendar (see `calendar()`).
plot

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>panel</td>
<td>A function in the form function(x, y, ...) which gives the action to be carried out in each panel of the display. The default is graphics::lines().</td>
</tr>
<tr>
<td>flip</td>
<td>A logical scalar: should the y-axis (ticks and numbering) be flipped from side 2 (left) to 4 (right) from series to series when facet is &quot;multiple&quot;?</td>
</tr>
<tr>
<td>ncol</td>
<td>An integer specifying the number of columns to use when facet is &quot;multiple&quot;. Defaults to 1 for up to 4 series, otherwise to 2.</td>
</tr>
<tr>
<td>xlab, ylab</td>
<td>A character vector giving the x and y axis labels.</td>
</tr>
<tr>
<td>main</td>
<td>A character string giving a main title for the plot.</td>
</tr>
<tr>
<td>sub</td>
<td>A character string giving a subtitle for the plot.</td>
</tr>
<tr>
<td>ann</td>
<td>A logical scalar: should the default annotation (title and x and y axis labels) appear on the plot?</td>
</tr>
<tr>
<td>axes</td>
<td>A logical scalar: should axes be drawn on the plot?</td>
</tr>
<tr>
<td>frame.plot</td>
<td>A logical scalar: should a box be drawn around the plot?</td>
</tr>
<tr>
<td>panel.first</td>
<td>An expression to be evaluated after the plot axes are set up but before any plotting takes place. This can be useful for drawing background grids.</td>
</tr>
<tr>
<td>panel.last</td>
<td>An expression to be evaluated after plotting has taken place but before the axes, title and box are added.</td>
</tr>
<tr>
<td>...</td>
<td>Further parameters to be passed to panel (e.g. graphical parameters).</td>
</tr>
</tbody>
</table>

Value

plot() is called for its side-effects: it results in a graphic being displayed. Invisibly returns x.

Author(s)

N. Frerebeau

See Also

graphics::plot()

Other plotting tools: image(), year_axis()

Examples

```r
## Create 6 time-series of 50 observations
## Sampled every two years starting from 2000 BP
X <- series(
    object = matrix(rnorm(300), nrow = 50, ncol = 6),
    time = seq(2000, by = -2, length.out = 50),
    calendar = BP()
)

## Multiple
plot(X) # Default calendar
plot(X, calendar = BP(), flip = TRUE) # BP
plot(X, calendar = b2k(), ncol = 1) # b2k
```
## Single
plot(X, facet = "single") # CE
plot(X, facet = "single", calendar = BP()) # BP

## Create 6 x 3 time-series of 50 observations
## Sampled every two years starting from 2000 BP
X <- series(
    object = array(rnorm(900), dim = c(50, 6, 3)),
    time = seq(2000, by = 2, length.out = 50),
    calendar = BP()
)
plot(X, calendar = BP(), flip = TRUE) # BP
plot(X, calendar = b2k(), ncol = 1) # b2k

## Graphical parameters
plot(X, lwd = c(1, 2, 3), col = c("#004488", "#DDAA33", "#BB5566"))
plot(X, type = "b", pch = 16, col = c("#004488", "#DDAA33", "#BB5566"))
plot(X, type = "p", pch = c(16, 17, 18), cex = c(1, 2, 3))

---

Pretty Breakpoints

### Description
Pretty Breakpoints

#### Usage

```
## S4 method for signature 'RataDie'
pretty(x, calendar = getOption("aion.calendar"), ...)  
```

### Arguments

- `x` A `RataDie` object.
- `calendar` A `TimeScale` object specifying the target calendar (see `calendar()`).
- `...` Further parameters to be passed to `base::pretty()`.

### Details

`pretty()` computes a vector of increasing numbers which are "pretty" in decimal notation of calendar. Pretty breakpoints are then converted to `rata die`.

### Value

A `RataDie` object.

### See Also

Other fixed date tools: `arithmetic`, `as_date`, `as_decimal`, `as_fixed`, `as_year`, `fixed`, `fixed_gregorian`, `fixed_julian`, `format()`
RataDie-class

RataDie-class  RataDie

Description

An S4 class to represent a vector of *rata die*.

Details

*Rata die* (fixed date) are represented as the number of days since 01-01-01 (Gregorian), with negative values for earlier dates.

It is intended that the date should be an integer value, but this is not enforced in the internal representation.

Slots

.Data A numeric vector giving the *rata die* values.

Note

This class inherits from numeric.

Author(s)

N. Frerebeau

See Also

Other classes: AD-class, BC-class, BCE-class, BP-class, CE-class, GregorianCalendar-class, JulianCalendar-class, TimeScale-class, TimeSeries-class, b2k-class

Other time classes: TimeSeries-class

series  Create Time Series

Description

Create Time Series
Usage

series(object, time, calendar, ...)  

## S4 method for signature 'array,RataDie,missing'
series(object, time, names = NULL)

## S4 method for signature 'array,numeric,TimeScale'
series(object, time, calendar, scale = 1, names = NULL)

## S4 method for signature 'matrix,numeric,TimeScale'
series(object, time, calendar, scale = 1, names = NULL)

## S4 method for signature 'matrix,RataDie,missing'
series(object, time, names = NULL)

## S4 method for signature 'numeric,numeric,TimeScale'
series(object, time, calendar, scale = 1, names = NULL)

## S4 method for signature 'numeric,RataDie,missing'
series(object, time, names = NULL)

## S4 method for signature 'data.frame,numeric,TimeScale'
series(object, time, calendar, scale = 1, names = NULL)

## S4 method for signature 'data.frame,RataDie,missing'
series(object, time, names = NULL)

Arguments

object       A numeric vector, matrix or array of the observed time-series values. A data.frame will be coerced to a numeric matrix via data.matrix().
time         A numeric vector of (decimal) years or a RataDie object (see fixed()).
calendar     A TimeScale object specifying the calendar of time (see calendar()). If missing, time must be a RataDie object.
...          Currently not used.
names        A character string specifying the names of the time series.
scale        A length-one numeric vector specifying the number of years represented by one unit. It should be a power of 10 (i.e. 1000 means ka).

Details

Data will be sorted in chronological order.

Value

A TimeSeries object.
**span**

**Author(s)**

N. Frerebeau

**See Also**

Other time series tools: `data.frame`, `span()`, `start()`, `time()`, `window()`

**Examples**

```r
## Create time-series of 20 observations

## Univariate
## Sampled every years starting from 1029 BCE
(X <- series(rnorm(30), time = 1029:1000, calendar = BCE()))

## Terminal and sampling times (returns rata die)
start(X)
end(X)
time(X)
span(X)

## Multivariate
## Sampled every century starting from 1000 CE
(Y <- series(matrix(rnorm(90), 30, 3), time = 1000:1029, calendar = CE()))

## Terminal and sampling times (returns Gregorian Common Era years)
start(Y, calendar = CE())
end(Y, calendar = CE())
time(Y, calendar = CE())
span(Y, calendar = CE())

## Coerce to data frame
df <- as.data.frame(Y, calendar = BP())
head(df)
```

---

### Description

Get the duration.

### Usage

```r
span(x, ...)
```

## S4 method for signature 'TimeSeries'

```r
span(x, calendar = NULL)
```
Arguments

- **x**: A *TimeSeries* object.
- **...**: Currently not used.
- **calendar**: A *TimeScale* object specifying the target calendar (see `calendar()`). If NULL (the default), *rata die* are returned.

Value

A numeric vector.

Author(s)

N. Frerebeau

See Also

Other time series tools: `data.frame`, `series()`, `start()`, `time()`, `window()`

Examples

```r
## Create time-series of 20 observations
## Univariate
## Sampled every years starting from 1029 BCE
(X <- series(rnorm(30), time = 1029:1000, calendar = BCE()))

## Terminal and sampling times (returns rata die)
start(X)
end(X)
time(X)
span(X)

## Multivariate
## Sampled every century starting from 1000 CE
(Y <- series(matrix(rnorm(90), 30, 3), time = 1000:1029, calendar = CE()))

## Terminal and sampling times (returns Gregorian Common Era years)
start(Y, calendar = CE())
end(Y, calendar = CE())
time(Y, calendar = CE())
span(Y, calendar = CE())

## Coerce to data frame
df <- as.data.frame(Y, calendar = BP())
head(df)
```
Description

Get the times the first and last observations were taken.

Usage

```r
## S4 method for signature 'TimeSeries'
start(x, calendar = NULL)
## S4 method for signature 'TimeSeries'
end(x, calendar = NULL)
```

Arguments

- `x`: A `TimeSeries` object.
- `calendar`: A `TimeScale` object specifying the target calendar (see `calendar()`). If `NULL` (the default), *rata die* are returned.

Value

A numeric vector.

Author(s)

N. Frerebeau

See Also

Other time series tools: `data.frame`, `series()`, `span()`, `time()`, `window()`

Examples

```r
## Create time-series of 20 observations

## Univariate
## Sampled every years starting from 1029 BCE
(X <- series(rnorm(30), time = 1029:1000, calendar = BCE()))

## Terminal and sampling times (returns rata die)
start(X)
end(X)
time(X)
span(X)

## Multivariate
## Sampled every century starting from 1000 CE
```
(Y <- series(matrix(rnorm(90), 30, 3), time = 1000:1029, calendar = CE()))

## Terminal and sampling times (returns Gregorian Common Era years)
start(Y, calendar = CE())
end(Y, calendar = CE())
time(Y, calendar = CE())
span(Y, calendar = CE())

## Coerce to data frame
df <- as.data.frame(Y, calendar = BP())
head(df)

---

## subset

**Extract or Replace Parts of an Object**

### Description
Operators acting on objects to extract or replace parts.

### Usage

```r
## S4 method for signature 'RataDie'
x[i]

## S4 method for signature 'TimeSeries'
x[i, j, k, drop = FALSE]
```

### Arguments

- `x`: An object from which to extract element(s) or in which to replace element(s).
- `i, j, k`: Indices specifying elements to extract or replace.
- `drop`: A `logical` scalar: should the result be coerced to the lowest possible dimension? This only works for extracting elements, not for the replacement.

### Value
A subsetted object.

### Author(s)
N. Frerebeau
Sampling Times of Time Series

Description

Get the sampling times:

- `time()` creates the vector of times at which a time series was sampled.
- `frequency()` returns the mean number of samples per unit time.

Usage

```r
## S4 method for signature 'TimeSeries'
time(x, calendar = NULL)
## S4 method for signature 'TimeSeries'
frequency(x)
```

Arguments

- `x`: A `TimeSeries` object.
- `calendar`: A `TimeScale` object specifying the target calendar (see `calendar()`). If `NULL` (the default), `rata die` are returned.

Value

A numeric vector.

Author(s)

N. Frerebeau

See Also

Other time series tools: `data.frame.series()`, `span()`, `start()`, `window()`

Examples

```r
## Create time-series of 20 observations

## Univariate
## Sampled every years starting from 1029 BCE
(X <- series(rnorm(30), time = 1029:1000, calendar = BCE()))

## Terminal and sampling times (returns rata die)
start(X)
end(X)
time(X)
```
## Multivariate
## Sampled every century starting from 1000 CE
(Y <- series(matrix(rnorm(90), 30, 3), time = 1000:1029, calendar = CE()))

## Terminal and sampling times (returns Gregorian Common Era years)
start(Y, calendar = CE())
end(Y, calendar = CE())
time(Y, calendar = CE())
span(Y, calendar = CE())

## Coerce to data frame
df <- as.data.frame(Y, calendar = BP())
head(df)

---

### TimeScale-class

#### Description

A virtual S4 class to represent a calendar.

#### Slots

- **label**: A character string specifying the abbreviated label of the time scale.
- **name**: A character string specifying the name of the time scale.
- **epoch**: A numeric value specifying the epoch year from which years are counted (starting date of the calendar, in years). Allows to define multiple era of a calendar.
- **fixed**: A numeric value specifying the reference date of the calendar (in rata die).
- **direction**: An integer specifying if years are counted backwards (-1) or forwards (1) from epoch.
- **year**: A numeric value giving the average length of the year in solar days.

#### Author(s)

N. Frerebeau

#### See Also

Other classes: `AD-class`, `BC-class`, `BCE-class`, `BP-class`, `CE-class`, `GregorianCalendar-class`, `JulianCalendar-class`, `RataDie-class`, `TimeSeries-class`, `b2k-class`

Other calendar classes: `GregorianCalendar-class`, `JulianCalendar-class`
Description
An S4 class to represent time series.

Details
A time series object is an \( nxmxp \) array, with \( n \) being the number of observations, \( m \) being the number of series and with the \( p \) columns of the third dimension containing extra variables for each series.

Slots
.Data A \( nxmxp \) numeric array giving the observed time-series values.
.Time A length-\( n \) RataDie object.

Note
This class inherits from array.

Author(s)
N. Frerebeau

See Also
Other classes: AD-class, BC-class, BCE-class, BP-class, CE-class, GregorianCalendar-class, JulianCalendar-class, RataDie-class, TimeScale-class, b2k-class
Other time classes: RataDie-class

window

Description
Extracts the subset of the object \( x \) observed between the times \( \text{start} \) and \( \text{end} \) (expressed in \text{rata die}).

Usage
```r
## S4 method for signature 'TimeSeries'
window(x, start = NULL, end = NULL)
```
Arguments

x  A TimeSeries object.
start  A length-one numeric vector specifying the start time of the period of interest.
end  A length-one numeric vector specifying the end time of the period of interest.

Value

A TimeSeries object.

Author(s)

N. Frerebeau

See Also

Other time series tools: data.frame, series(), span(), start(), time()

Examples

## Create 3 time-series of 100 observations
## Sampled every years starting from 1000 CE
(x <- series(matrix(rnorm(300), 100, 3), time = 1000:1099, calendar = CE()))

## Subset between 1025 and 1050 CE
(y <- window(x, start = 374009, end = 383140))

year_axis

Time Series Plotting Functions

Description

Time Series Plotting Functions

Usage

year_axis(
  side,
  at = NULL,
  format = c("a", "ka", "Ma", "Ga"),
  labels = TRUE,
  calendar = getOption("aion.last_calendar"),
  current_calendar = getOption("aion.last_calendar"),
  ...
)

year_axis

Arguments

side
   An integer specifying which side of the plot the axis is to be drawn on. The
   axis is placed as follows: 1=below, 2=left, 3=above and 4=right.

at
   A numeric vector giving the points at which tick-marks are to be drawn. If
   NULL, tickmark locations are computed.

format
   A character string specifying the prefix. It should be one of "a", "ka", "Ma" or
   "Ga". If TRUE, a good guess for an appropriate format is made.

labels
   A logical scalar specifying whether annotations are to be made at the tick-
   marks, or a vector of character strings to be placed at the tickpoints.

calendar
   A TimeScale object specifying the target calendar (see calendar()).

current_calendar
   A TimeScale object specifying the calendar used by the last call to plot().

... Further parameters to be passed to graphics::axis(). (e.g. graphical param-
eters).

Value

year_axis() is called it for its side-effects.

Author(s)

N. Frerebeau

See Also

Other plotting tools: image(), plot()

Examples

## Create a time-series of 300 observations
## Sampled every two years starting from 2000 BP
X <- series(
   object = rnorm(300),
   time = seq(2000, by = -2, length.out = 300),
   calendar = BP()
)

## Axis
plot(X, axes = FALSE, calendar = BP()) # Remove axes
year_axis(side = 1) # Same calendar as last plot
year_axis(side = 3, calendar = CE()) # Specific calendar
mtext(format(CE()), side = 3, line = 3)

## Grid
plot(X, panel.first = graphics::grid())
Index

* calendar classes
  TimeScale-class, 34

* calendar tools
  calendar, 9
  calendar_get, 10
  convert, 12
  gregorian, 20
  is, 22
  julian, 23

* classes
  RataDie-class, 27
  TimeScale-class, 34
  TimeSeries-class, 35

* datasets
  dates, 14

* fixed date tools
  arithmetic, 3
  as_date, 4
  as_decimal, 5
  as_fixed, 6
  as_year, 7
  fixed, 15
  fixed_gregorian, 16
  fixed_julian, 18
  format, 19
  pretty, 26

* mutators
  subset, 32

* plotting tools
  image, 21
  plot, 24
  year_axis, 36

* time classes
  RataDie-class, 27
  TimeSeries-class, 35

* time series tools
  data.frame, 13
  series, 27
  span, 29

  start, 31
  time, 33
  window, 35
  .RataDie(RataDie-class), 27
  .TimeScale(TimeScale-class), 34
  .TimeSeries(TimeSeries-class), 35
  [,RataDie-method (subset), 32
  [,TimeSeries-method (subset), 32

  AD(gregorian), 20
  Arith,numeric,RataDie-method
    (arithmetic), 3
  Arith,RataDie,numeric-method
    (arithmetic), 3
  Arith,RataDie,RataDie-method
    (arithmetic), 3
  arithmetic, 3, 4, 6–8, 16, 17, 19, 20, 26
  array, 35
  as.data.frame,TimeSeries-method
    (data.frame), 13
  as_date, 3, 4, 6–8, 16, 17, 19, 20, 26
  as_date,numeric,GregorianCalendar-method
    (as_date), 4
  as_date,numeric,JulianCalendar-method
    (as_date), 4
  as_date-method (as_date), 4
  as_decimal, 3, 4, 5, 7, 8, 16, 17, 19, 20, 26
  as_decimal,numeric,numeric,numeric,TimeScale-method
    (as_decimal), 5
  as_decimal-method (as_decimal), 5
  as_fixed, 3, 4, 6, 6, 8, 16, 17, 19, 20, 26
  as_fixed,numeric-method (as_fixed), 6
  as_fixed-method (as_fixed), 6
  as_year, 3, 4, 6, 7, 7, 16, 17, 19, 20, 26
  as_year,numeric,GregorianCalendar-method
    (as_year), 7
  as_year,numeric,JulianCalendar-method
    (as_year), 7
  as_year-method (as_year), 7
INDEX

b2k (gregorian), 20
base::pretty(), 26
BC (gregorian), 20
BCE (gregorian), 20
BP (gregorian), 20
dates, 14
dates, start, TimeSeries-method (start), 31
dates, start-method (start), 31
era::era(), 9
fixed, 3, 4, 6–8, 15, 17, 19, 20, 26
fixed(), 4, 8, 17, 18, 28
fixed, numeric, missing, missing, GregorianCalendar-method (fixed), 15
fixed, numeric, numeric, numeric, GregorianCalendar-method (fixed), 15
fixed, numeric, numeric, numeric, JulianCalendar-method (fixed), 15
fixed-method (fixed), 15
fixed_from_AD (fixed_gregorian), 16
fixed_from_b2k (fixed_gregorian), 16
fixed_from_BC (fixed_gregorian), 16
fixed_from_BCE (fixed_gregorian), 16
fixed_from_BP (fixed_gregorian), 16
fixed_from_CE (fixed_gregorian), 16
fixed_from_julian (fixed_julian), 18
fixed_gregorian, 3, 4, 6–8, 16, 19, 20, 26
fixed_julian, 3, 4, 6–8, 16, 17, 18, 20, 26
fixed_to_AD (fixed_gregorian), 16
fixed_to_b2k (fixed_gregorian), 16
fixed_to_BC (fixed_gregorian), 16
fixed_to_BCE (fixed_gregorian), 16
fixed_to_BP (fixed_gregorian), 16
fixed_to_CE (fixed_gregorian), 16
fixed_to_julian (fixed_julian), 18
format, GregorianCalendar, 9
format, numeric, numeric, numeric, GregorianCalendar-method (fixed), 15
format, numeric, numeric, numeric, JulianCalendar-method (fixed), 15
function, 12, 25
g graphical parameters, 25, 37
g graphics::axis(), 37
g graphics::image(), 22
g graphics::lines(), 25
g graphics::plot(), 25
ggregorian, 9, 11, 12, 20, 23
GregorianCalendar, 21
image, 21, 25, 37
image, TimeSeries-method (image), 21
integer, 11, 15, 22, 25, 34, 37
calendar, 9, 11, 12, 21, 23
calendar(), 4, 5, 8, 13, 15, 20–24, 26, 28, 30, 31, 33, 37
calendar, character-method (calendar), 9
calendar-method (calendar), 9
calendar_direction (calendar_get), 10
calendar_direction, TimeScale-method (calendar_get), 10
calendar_direction-method (calendar_get), 10
calendar_epoch (calendar_get), 10
calendar_epoch, TimeScale-method (calendar_get), 10
calendar_epoch-method (calendar_get), 10
calendar_fixed (calendar_get), 10
calendar_fixed, TimeScale-method (calendar_get), 10
calendar_fixed-method (calendar_get), 10
calendar_get (calendar_get), 10
calendar_label (calendar_get), 10
calendar_label, TimeScale-method (calendar_get), 10
calendar_label-method (calendar_get), 10
calendar_name (calendar_get), 10
calendar_name, TimeScale-method (calendar_get), 10
calendar_name-method (calendar_get), 10
calendar_unit (calendar_get), 10
calendar_unit, TimeScale-method (calendar_get), 10
calendar_unit-method (calendar_get), 10
CE (gregorian), 20
calendar, character, 9, 11, 19, 20, 24, 25, 28, 34, 37
convert, 9, 11, 12, 21, 23
convert, character, character-method (convert), 12
convert, TimeScale, TimeScale-method (convert), 12
convert-method (convert), 12
data.frame, 13, 14, 28–31, 33, 36
data.frame(), 13
data.matrix(), 28
INDEX

is, 9, 11, 12, 21, 22, 23
is_gregorian (is), 22
is_gregorian, ANY-method (is), 22
is_gregorian-method (is), 22
is_julian (is), 22
is_julian, ANY-method (is), 22
is_julian-method (is), 22
J (julian), 23
julian, 9, 11, 12, 21, 23, 23
JulianCalendar, 23
logical, 3, 8, 20, 23, 25, 32, 37
numeric, 3–6, 8, 11, 15, 17, 18, 27, 28, 30, 31, 33, 34, 36, 37
plot, 22, 24, 37
plot(), 37
plot, TimeSeries, missing-method (plot), 24
pretty, 3, 4, 6–8, 16, 17, 19, 20, 26
pretty, RataDie-method (pretty), 26
RataDie, 3, 4, 6, 8, 15, 17–19, 26, 28, 35
RataDie-class, 27
series, 13, 27, 30, 31, 33, 36
series, array, numeric, TimeScale-method (series), 27
series, array, RataDie, missing-method (series), 27
series, data.frame, numeric, TimeScale-method (series), 27
series, data.frame, RataDie, missing-method (series), 27
series, matrix, numeric, TimeScale-method (series), 27
series, matrix, RataDie, missing-method (series), 27
series, numeric, numeric, TimeScale-method (series), 27
series, numeric, RataDie, missing-method (series), 27
series-method (series), 27
sign, 11
span, 13, 29, 29, 31, 33, 36
span, TimeSeries-method (span), 29
span-method (span), 29
start, 13, 29, 30, 31, 33, 36
start, TimeSeries-method (start), 31
start-method (start), 31
subset, 32
time, 13, 29–31, 33, 36
time, TimeSeries-method (time), 33
time-method (time), 33
TimeScale, 4, 5, 8, 9, 11–13, 15, 20, 22, 24, 26, 28, 30, 31, 33, 37
TimeScale-class, 34
TimeSeries, 13, 22, 24, 28, 30, 31, 33, 36
TimeSeries-class, 35
window, 13, 29–31, 33, 35
window, TimeSeries-method (window), 35
window-method (window), 35
year_axis, 22, 25, 36