Package 'tolBasis'

November 5, 2015

Type Package

Title Fundamental Definitions and Utilities of the Time Oriented Language (TOL)

Version 1.0

Date 2015-11-02

Author Pedro Gea

Maintainer Pedro Gea <pgea@bayesforecast.com>

Description

Imports the fundamental definitions and utilities of the Time Oriented Language (TOL), focused on time series analysis and stochastic processes, and provides the basis for the integration of TOL in R. See <https://www.tol-project.org> for more information about the TOL project.

URL https://www.tol-project.org/browser/tolp/Rprojects/tolBasis

License GPL-3

Depends lubridate

Imports polynom

Suggests xts, zoo

NeedsCompilation no

Repository CRAN

Date/Publication 2015-11-05 13:38:39

R topics documented:

`tolBasis-package` *TOL Basis*

Description

`tolBasis` provides basis for the integration of TOL in R.

Details

Package:	tolBasis
Type:	Package
Version:	1.0
Date:	2015-03-02
License:	GPL-3

Author(s)

Pedro Gea

Maintainer: Pedro Gea <pgea@bayesforecast.com>

References

See the main page of the TOL-Project: https://www.tol-project.org

See Also

See the main classes implemented:

- Date: the R class Date used for TOL integration
- Dating: a practical version of the TOL grammar TimeSet
- Polyn: an implementation of the TOL grammar Polyn
- Ratio: an implementation of the TOL grammar Ratio
- Serie: a limited implementation of the TOL grammar Serie

---

as.Polyn *Polyn Conversion Functions*

Description

Function to convert between different objects to class "Polyn" representing lag polynomials as in TOL language.

Usage

```
as.Polyn(x, ...)
```

Arguments

x	object to be coerced
...	further arguments passed to or from other methods

Value

Returns an object of class "Polyn".

Examples

```
# Creates lag polynomial: 1+B^2
p <- as.Polyn(c(1,0,1))
```

`as.Ratio` *Ratio Conversion Functions*

Description

Function to convert between different objects to class "Ratio" representing lag polynomials quotient as in TOL language.

Usage

```
as.Ratio(x, ...)
```

Arguments

`x`	object to be coerced
`...`	further arguments passed to or from other methods

Value

Returns an object of class "Ratio".

Examples

```
# Creates lag polynomial quotient: (1+B^2) / (1)
p <- as.Ratio(1+B^2)
```

`as.Serie` *Serie Compatibility*

Description

Converts a Time-Series (ts) object to a Serie one.

Usage

```
as.Serie(x, ...)
```

Arguments

x	a Time-Series (ts) object
...	further arguments

Value

a Series object

Examples

```
ts1 <- ts(1:10, frequency = 4, start = c(1995, 2))
s1 <- as.Serie(ts1)
```

---

as.ts *Time-Series (ts) Compatibility*

Description

Converts a Serie object to a Time-Series (ts) one.

Usage

```
## S3 method for class 'Serie'
as.ts(x, ...)
```

Arguments

x	a Serie object
...	further arguments (unused with Serie objects)

Value

a Time-Series (ts) object

See Also

See also ts

Examples

```
s1 <- Serie(rnorm(12), Monthly, as.Date(ymd("2001-01-01")))
ts1 <- as.ts(s1)
```

as.xts *eXtensible Time-Series (xts) Compatibility*

Description

Converts a Serie object to an eXtensible Time-Series (xts) one.

Usage

```
## S3 method for class 'Serie'
as.xts(x, ...)
```

Arguments

x	a Serie object
...	further arguments (unused with Serie objects)

Value

an eXtensible Time-Series (xts) object

See Also

See also xts

Examples

```
## Not run:
library(xts)
s1 <- Serie(rnorm(12), Monthly, as.Date(ymd("2001-01-01")))
xts1 <- as.xts(s1)

## End(Not run)
```

Date *Date Class*

Description

Uses the R class Date for the TOL integration in R.

Usage

```
# Creates a date using year-month-day specification.
Date(year, month=1, day=1)
```

Arguments

year	a valid year
month	the number of a month
day	a valid day of a month

Value

a Date object

See Also

See also the reference for Date objects.

Examples

```
dt <- Date(2015) #"2015-01-01"
```

DateSet *Dating DateSet*

Description

Creates a new dating of subclass DateSet.

Usage

```
DateSet(name, dates, envir=.Dating, overwrite=FALSE)
```

Arguments

name	a valid Dating name (not yet in use)
dates	a set of all dates belonging to the dating
envir	an environment to be used
overwrite	indicates whether an existing Dating should be overwritten

Details

An environment called .Dating is defined and used by default to contain the user-defined datesets.

Value

Returns the new Dating object. The object and its dates (called <name>.Dates) are assigned at the specified environment.

See Also

See also the reference for Dating objects.

Examples

```
# Creates a new dateset with an only date: today
## Not run: DateSet("Today", Sys.Date())
```

`Dating` *Dating Class*

Description

Implements the TOL grammar "TimeSet" (Dating) in R.

Usage

```
## S3 method for class 'Dating'
print(x, ...)
```

Arguments

`x`	a Dating object
`...`	further arguments

Value

The method `print` prints the name of the Dating object and returns it.

See Also

The predefined datings are:

- Standard Datings: `Yearly`, `Monthly`, `Weekly`, `Daily`
- Weekdays Datings: `Mondays`, ..., `Sundays`
- N-Monthly Datings: `Quarterly`, `HalfYearly`

See also the function `DateSet` to create custom datings.

See the generic function `print`

Dbelong *Date Belong*

Description

Indicates if a date belongs to a dating.

Usage

```
Dbelong(dte, dating)
```

Arguments

dte	a Date or POSIXt object
dating	a Dating object

Value

Returns a logical value depending on if the date belongs or not to the dating.

See Also

See also the reference for Dating objects.

Examples

```
# Checks if today is the first day of a month
Dbelong(Sys.Date(), Monthly)
```

Dceiling *Date Ceiling*

Description

Date Ceiling in a Dating

Usage

```
Dceiling(dte, dating=Daily)
```

Arguments

dte	a Date or POSIXt object
dating	a Dating object

Value

Returns the least date belonging to the dating which is greater or equal than the indicated one.

See Also

See also the functions `Dfloor` and `Dround`

Examples

```
# Obtain the first day of the next month
Dceiling(Sys.Date(), Monthly)
```

`Dcheck` *Check a Date sequence*

Description

Checks if a Date sequence is compatible with the specified Dating object.

Usage

```
Dcheck(date.sequence, dating)
```

Arguments

`date.sequence`	a sequence of dates (class Date or POSIXt)
`dating`	a Dating object

Value

Returns whether the Date sequence corresponds with a Date sequence at the specified Dating object.

Note that the dates should be ordered in order to be a valid sequence.

See Also

See also the functions `Dseq` and `Dfind`

Examples

```
# Check a Date sequence
date.sequence <- Dseq(Date(2010), Date(2014,12), Monthly)
Dcheck(date.sequence, Monthly) #-> TRUE
Dcheck(date.sequence, Quarterly) #-> FALSE
```

Ddiff *Date Difference*

Description

Date Difference in a Dating

Usage

```
Ddiff(dte1, dte2, dating=Daily)
```

Arguments

dte1	a Date or POSIXt object
dte2	a Date or POSIXt object
dating	a Dating object

Value

Returns the difference in dates belonging to the dating between the indicated dates. If the dates do not belong to the dating the floor dates are used instead.

Examples

```
# Obtain how many days until the next new year day.
dt1 <- Sys.Date()
dt2 <- Dsucc(dt1, Yearly, 1)
Ddiff(dt1, dt2, Daily)
```

Dfind *Find Dating for a Date sequence*

Description

Finds a Dating object compatible with a Date sequence.

Usage

```
Dfind(date.sequence)
```

Arguments

date.sequence	a sequence of dates (class Date or POSIXt)

Value

Returns (if it is found) a Dating compatible with the specified Date sequence.

Note that the dates should be ordered in order to be a valid sequence.

See Also

See also the functions Dseq and Dcheck

Examples

```
# Check a sequence of dates
Dfind(c(Date(2010), Date(2010,2), Date(2010,3))) #-> Monthly
```

Dfloor *Date Floor*

Description

Date Floor in a Dating

Usage

```
Dfloor(dte, dating=Daily)
```

Arguments

dte	a Date or POSIXt object
dating	a Dating object

Value

Returns the greatest date belonging to the dating which is less or equal than the indicated one.

See Also

See also the functions Dceiling and Dround

Examples

```
# Obtain the first day of the current month
Dfloor(Sys.Date(), Monthly)
```

Droud *Date Round*

Description

Date Round in a Dating

Usage

```
Dround(dte, dating=Daily)
```

Arguments

dte	a Date or POSIXt object
dating	a Dating object

Value

Returns the nearest date to the indicated one that belongs to the dating.

See Also

See also the functions Dfloor and Dceiling

Examples

```
# Obtain the nearest first day of a month
Dround(Sys.Date(), Monthly)
```

Dseq *Date Sequence*

Description

Date sequence in a Dating

Usage

```
Dseq(from, to, dating, len)
# Sequence of dates in an interval
# Dseq(from, to, dating, )
# Sequence of dates from a date
# Dseq(from, , dating, lenght.out)
```

Arguments

from	the Date o POSIXt objet indicating the minimum date
to	the Date o POSIXt objet indicating the maximum date
dating	a Dating object
len	the number of obtained dates whether the argument to is missing

Value

Returns a vector of dates belonging to the dating in the interval [from, to] If the argument to is missing returns a vector with the length 'len'.

Examples

```
# Vector of the monthly dates of the current year
currYear <- Dfloor(Sys.Date(), Yearly)
Dseq(currYear, dating=Monthly, len=12)
```

Dsucc *Date Successor*

Description

Date successor in a dating.

Usage

```
Dsucc(dte, dating=Daily, num=1)
```

Arguments

dte	a Date or POSIXt object
dating	a Dating object
num	the number of successive dates

Value

Returns the corresponding successive date.

See Also

See also Dfloor, Dceiling and Dround.

Examples

```
dt1 <- Sys.Date()
dt2 <- Dsucc(dt1, Monthly, 2)
```

`Lag Operator` *Lag Operators*

Description

Lag operator or backward operator: `B`.

Inverse lag operator or forward operator: `A`.

See Also

See the reference to Polyn class.

Examples

```
p <- 1 + B
```

`N-Monthly Datings` *N-Monthly Datings*

Description

The dates of N-Monthly datings are a subset of the Monthly dates with a fixed periodicity (N) from January onwards.

Defined N-Monthly Datings:

- `Quarterly`: The dating containing all the first day of a quarter or trimester. I. e. 1st january, 1st April, 1st July and 1st October.
- `HalfYearly`: The dating containing all the first day of a half-year. I. e. 1st January and 1st July.

See Also

See also the standard datings `Monthly` and `Yearly`

operator %:% *Polyn-Serie Operator*

Description

Lags a time series with an lag polynomial.

Usage

```
p %:% s
```

Arguments

p	a Polyn object
s	a Serie object

Value

a Serie object

Examples

```
s <- Serie(rnorm(12), Monthly, as.Date(ymd("2001-01-01")))
sd <- (1+B) %:% s
```

plot.Serie *Serie Plotting*

Description

Plotting method for time-series inheriting from class "Serie".

Usage

```
## S3 method for class 'Serie'
plot(x, y, ...,
  from, to, ylim, dating, date.format,
  axes=c(T,T,F,F), legend.names, style)
```

Arguments

x, y	Serie objects.
...	more Serie objects or additional graphical arguments.
from, to	dates from and to which x axis should be plotted. They are used instead the argument xlim. See plot.window
ylim	numeric vector of length 2, giving the y coordinates range. See plot.window
dating	Dating object used to represent tickmarks at x axis. By default it is selected from the commonest datings according to the x value of the lab argument.
date.format	A character string. The default format depends on the selected dating. The standard format is "%Y-%m-%d". See format.POSIXlt.
axes	a logical vector indicating which axes should be drawn.
legend.names	names of the lines that should be shown in the legend. Use a list with length 0 for show default values. See legend.
style	name of a style configuration that defines a set of graphical options. See section Plot Styles.

Details

This plotting method uses internally other graphics methods as: box, axis or line.

Common graphical arguments as col, lwd or lty can be redirected to the corresponding sub-method with one of the following prefixes:

box. Prefix for drawing the box.

axes. Prefix for drawing the axes. A vector or a list of options can be used in order to use different options for different axes.

lines. Prefix for drawing the lines. A vector or a list of options can be used in order to use different options for different lines. The values are used cyclely if there are more lines than options.

Methods title and legend are also internally called. Use the corresponding prefixes to specify their subset of arguments:

title. Prefix for adding titles and labels. For example: title.main. Note that the graphical properties should be specified as usual, for example: col.main. See par.

legend. Prefix for customizing the legend. For example, the label color should be specified with legend.text.col. Note, that the legend is drawn when argument legend.names is not missing.

Plot Styles

Currently, only the configuration that imitates TOL plots (style="TOL") is defined. The configuration is:

```
lab=c(10,5,7), bty="?", las=2, xaxs="i", yaxs="i", lines.lwd=2
```

When argument bty is assigned as "?", it is replaced according to the axes, drawing a partial box. For example, when the commonest axes ("bottom" and "left") are used, a box "L" is drawn.

Usual Graphical Arguments

`bg` background color

bty a character string which determined the type of box which is drawn. A value of "n" suppresses the box.

`col` specification for the default plotting color. See section 'Color Specification' at `par`.

`lines.col` `col` in combination with prefix `line.` indicates colors used to draw the lines. By default, usual TOL colors are used. Other lists as `rainbow(...)` or `heat.colors(...)` can be used. See Palettes.

`col.axis` The color to be used for axis annotation.

`col.lab` The color to be used for x and y labels.

`col.main, col.sub` Colors to be used for plot main titles and sub-titles.

lab A numerical vector of the form `c(x, y, len)` which modifies the default way that axes are annotated. The values of `x` and `y` give the (approximate) number of tickmarks on the x and y axes and `len` specifies the label length. The value for x axis is used to determine the Dating object used to represent the tickmarks. See argument `dating`.

las numeric in 0,1,2,3 indicating the style of axis labels: parallel, horizontal, perpendicular or vertical.

`lwd` The line width, a positive number, defaulting to 1. Different widths can be used via `box.lwd`, `axes.lwd` and `lines.lwd`.

`lty` The line type. Line types can either be specified as an integer (0=blank, 1=solid (default), 2=dashed, 3=dotted, 4=dotdash, 5=longdash, 6=twodash) or as a valid character string. See section 'Line Type Specification' at `par`.

`mar` A numerical vector of the form `c(bottom, left, top, right)` which gives the number of lines of margin to be specified on the four sides of the plot. The default is `c(5, 4, 4, 2) + 0.1`. Changing these values may be convenient when title or legend are shown.

`mgp` The margin line (in `mex` units) for the axis title, axis labels and axis line. Note that `mgp[1]` affects `title` whereas `mgp[2:3]` affect `axis`. The default is `c(3, 1, 0)`. In combination with `axes.` different options can be specified for each axis.

xaxs, yaxs style of axis interval calculation to be used for the x and y axes. Main values are "r" (regular) or "i" (internal).

See Also

See `par` for more details or more graphical options.

See also `plot`, `plot.default`.

Examples

```
s1 <- Serie(rnorm(18), Monthly, Date(2001))
s2 <- Serie(rnorm(12), Monthly, Date(2002))
plot(s1, s2)
plot(s1, s2, style="TOL")
plot(s1, s2, from=Date(2001,7), dating=Monthly,
  style="TOL", axes.lwd=2, bg="snow2",
  axes.font.axis=c(3,1), title.main="Example")
```

```
plot(s1, s2, s1-s2, dating=Quarterly, date.format="%m'%y",
  mar=c(3,3,2,2), lines.col=rainbow(5), lines.lwd=2)
plot(s1, s2, style="TOL", mar=c(6,4,2,2),
  legend.names=c("s1", "s2"))
plot(s1, s2, dating=Quarterly, date.format="%Y\n%m",
  axes.mgp=list(c(3,2,0),c(3,1,0)), las=1)
```

`Polyn` *Polyn Class*

Description

Implements the TOL grammar "Polyn" in R.

Usage

```
# Creates a new Polyn Object
Polyn(coeffs, base=0)
## S3 method for class 'Polyn'
as.character(x, ..., backward="B", forward="A")
## S3 method for class 'Polyn'
print(x, ...)
```

Arguments

`coeffs`	a vector of coefficients
`base`	the degree of the first coefficient
`x`	a Polyn object
`...`	further arguments
`backward`	the character to print the lag operator
`forward`	the character to print the inverse lag operator

Value

The method `print` prints the expression of the Polyn object and returns it.

See Also

See the generic function `print`

Examples

```
# Creates lag polynomial: 1+B^2
p <- Polyn(c(1,0,1))
# Gets the Polyn object expression as in TOL
as.character(A+B, forward="F")
# Prints the Polyn object as in TOL
print(A+B, forward="F")
```

`Polyn Arithmetic` *Polyn Arithmetic Operators*

Description

Arithmetic Operators for the "Polyn" class.

Usage

```
## S3 method for class 'Polyn'
p1 + p2
## S3 method for class 'Polyn'
p1 - p2
## S3 method for class 'Polyn'
p1 * p2
## S3 method for class 'Polyn'
p / x
## S3 method for class 'Polyn'
p ^ n
```

Arguments

`p1, p2, p`	Polyn objects
`x`	a real number or a Polyn object
`n`	a positive integer number

Value

Returns the Polyn object resulting of the arithmetical operation.

Note

If the argument `x` is a Polyn object, a Ratio object is returned.

See Also

See also `Ratio`

Examples

```
p1 <- 1+B
p2 <- (A+B)^2
p3 <- p1*p2
```

Polyn Coefficents *Polyn Coefficients*

Description

Returns a Polyn coefficient

Usage

```
## S3 method for class 'Polyn'
p[index, ..., degree]
```

Arguments

p	a Polyn objecy
index	index of the element
degree	degree of an element of the Polyn
...	unused arguments

Value

Returns the coefficient of the selected index or degree.

Examples

```
p <- (1+B)^2
p[1]
p[degree=2]
```

Polyn Comparison *Polyn Relational Operators*

Description

Relational operators for the Polyn class.

Usage

```
## S3 method for class 'Polyn'
p1 == p2
## S3 method for class 'Polyn'
p1 != p2
## S3 method for class 'Polyn'
p1 < p2
## S3 method for class 'Polyn'
```

```
p1 <= p2
## S3 method for class 'Polyn'
p1 > p2
## S3 method for class 'Polyn'
p1 >= p2
```

Arguments

p1, p2	Polyn objects

Value

Returns the value resulting of the relational operation. The order operators are not implemented and return NA.

Examples

```
1+B==B+1
```

Ratio *Ratio Class*

Description

Implements the TOL grammar "Ratio" in R.

Usage

```
Ratio(num, den)
## S3 method for class 'Ratio'
as.character(x, ...)
## S3 method for class 'Ratio'
print(x, ...)
```

Arguments

num, den, x	Ratio objects
...	further arguments

Value

The method `print` prints the expression of the Ratio object and returns it.

See Also

See the generic function `print` and the relative function for Polyn objects: `print.Polyn`.

Examples

```
# Gets the Ratio object expression as in TOL
as.character(Ratio(1, A+B), forward="F")
# Prints the Ratio object as in TOL
print(Ratio(1, A+B), forward="F")
```

`Ratio Arithmetic` *Ratio Arithmetic Operators*

Description

Arithmetic Operators for the "Ratio" class.

Usage

```
## S3 method for class 'Ratio'
r1 + r2
## S3 method for class 'Ratio'
r1 - r2
## S3 method for class 'Ratio'
r1 * r2
## S3 method for class 'Ratio'
r1 / r2
## S3 method for class 'Ratio'
r ^ n
```

Arguments

`r1, r2, r`	Ratio objects
`n`	a positive integer number

Value

Returns the Ratio object resulting of the arithmetical operation.

Examples

```
r1 <- Ratio(1, 1+B)
r2 <- Ratio(A-B, (A+B)^2)
r3 <- r1 - r2
```

Ratio Comparison *Ratio Relational Operators*

Description

Relational operators for the Ratio class.

Usage

```
## S3 method for class 'Ratio'
r1 == r2
## S3 method for class 'Ratio'
r1 != r2
## S3 method for class 'Ratio'
r1 < r2
## S3 method for class 'Ratio'
r1 <= r2
## S3 method for class 'Ratio'
r1 > r2
## S3 method for class 'Ratio'
r1 >= r2
```

Arguments

r1, r2	Ratio objects

Value

Returns the value resulting of the relational operation. The order operators are not implemented and return NA.

Examples

```
Ratio(1,1+B)==Ratio(1,B+1)
```

Rdenominator *Ratio Denominator*

Description

Denominator of a Ratio object

Usage

```
Rdenominator(r)
```

Arguments

r	a Ratio object

Value

Returns the Polyn object corresponding to the denominator of the Ratio.

See Also

See also the function Rnumerator

Examples

```
# Obtain the denominator of a Ratio object
ratio <- Ratio(1, 1+B) + Ratio(1, 1-B)
Rdenominator(ratio)
```

Rnumerator *Ratio Numerator*

Description

Numerator of a Ratio object

Usage

```
Rnumerator(r)
```

Arguments

r	a Ratio object

Value

Returns the Polyn object corresponding to the numerator of the Ratio.

See Also

See also the function Rdenominator

Examples

```
# Obtain the numerator of a Ratio object
ratio <- Ratio(1, 1+B) + Ratio(1, 1-B)
Rnumerator(ratio)
```

Sdates *Dates of a Serie*

Description

Obtain the dates of a time series.

Usage

```
Sdates(s)
```

Arguments

s	a Serie object

Value

a vector of dates

Examples

```
s <- Serie(rnorm(12), Monthly, Date(2015))
Sdates(s)
```

Sdating *Serie Dating*

Description

Obtains the dating of a time-series.

Usage

```
Sdating(s)
```

Arguments

s	a Serie object

Value

a Dating object

See Also

See also Dating

Examples

```
s <- Serie(rnorm(12), Monthly, as.Date(ymd("2001-01-01")))
Sdating(s)
```

`Serie` *Serie Class*

Description

Implements the TOL grammar "Serie" in R.

Usage

```
# Creates a new Serie Object
Serie(data, dating, begin)
## S3 method for class 'Serie'
print(x, ..., limit, mode)
```

Arguments

`data`	a numeric vector of data
`dating`	a Dating object
`begin`	a Date object
`x`	a Serie object
`...`	further arguments
`limit`	the maximum number of data printed; default value is controlled by the option `max.print.Serie`
`mode`	the mode in which limited data will be printed; if mode is less than zero first data are printed, if it is greater than zero last data are printed, if zero (symmetric mode) some data from the first data and from the last data are printed; default value is controlled by the option `mode.print.Serie`

Value

The function `Serie` creates a new time-series. Returns a Serie object.

The method `print` prints the expression of the Polyn object and returns it.

See Also

Serie class related items:

- Arithmetic operators
- Basic attributes: `Sdating`, `Sfirst`, `Slast`
- Other functions: `Ssub`

See the generic function `print`

See also the R time-series implementation: ts

Examples

```
s1 <- Serie(rnorm(12), Monthly, Date(2015))
```

`Serie Arithmetic` *Serie Arithmetic Operators*

Description

Arithmetic operators for the Serie class.

Usage

```
## S3 method for class 'Serie'
s1 + s2
## S3 method for class 'Serie'
s1 - s2
## S3 method for class 'Serie'
s1 * s2
## S3 method for class 'Serie'
s1 / s2
## S3 method for class 'Serie'
s ^ n
```

Arguments

`s1, s2, s`	Serie objects
`n`	a positive integer number

Value

Returns the Serie object resulting of the arithmetical operation.

See Also

See the class Serie.

Examples

```
s1 <- Serie(rnorm(12), Monthly, Date(2015))
s2 <- Serie(rnorm(12), Monthly, Date(2015))
ss <- s1+s2
sd <- s1-s2
sp <- s1*s2
sq <- s1/s2
se <- s1^2
```

Serie Coefficents *Serie Coefficients*

Description

Returns the coeffients of a time-series.

Usage

```
## S3 method for class 'Serie'
s[index]
```

Arguments

s	a Serie object
index	the index of an element or a valid date in the time-series dating

Value

Returns the coefficient of the selected index or date.

Examples

```
s <- Serie(rnorm(12), Monthly, Date(2015))
s[1]
s[Date(2015)]
```

Sfirst *First Date of a Serie*

Description

Obtains the first date of a time series.

Usage

```
Sfirst(s)
```

Arguments

s	a Serie object

Value

a Date object

See Also

See the reference to Serie class.

See also the method Slast.

Examples

```
s <- Serie(rnorm(12), Monthly, Date(2015))
Sfirst(s)
```

`Slast` *Last Date of a Serie*

Description

Obtains the last date of a time series.

Usage

```
Slast(s)
```

Arguments

`s`	a Serie object

Value

a Date object

See Also

See the reference to Serie class.

See also the method Sfirst.

Examples

```
s <- Serie(rnorm(12), Monthly, Date(2015))
Slast(s)
```

Ssub *Subset of a Serie*

Description

Extracts a time series in a sub-interval of the original time series.

Usage

```
Ssub(s, from=NA, to=NA)
```

Arguments

s	a Serie object
from	a Date object representing the first Date of the new time series
to	a Date object representing the last Date of the new time series

Value

a Serie object

Examples

```
s <- Serie(rnorm(12), Monthly, Date(2015))
ss <- Ssub(s, Date(2015,3), Date(2015,10))
```

Standard Datings *Standard Datings*

Description

Description of the implemented standard datings.

Standard Datings:

- `Daily`: The dating containing all the dates.
- `Weekly`: The dating containing all the sundays (first day of a week in R).
- `Monthly`: The dating containing all the first day of a month.
- `Yearly`: The dating containing all the first day of a year.

`Weekdays Datings` *Weekdays Datings*

Description

Description of the weekdays datings.

Weekdays Datings:

- `Mondays`: The dating containing all the mondays.
- `Tuesdays`: The dating containing all the tuesdays.
- `Wednesdays`: The dating containing all the wednesdays.
- `Thursdays`: The dating containing all the thursdays.
- `Fridays`: The dating containing all the fridays.
- `Saturdays`: The dating containing all the saturdays.
- `Sundays`: The dating containing all the sundays. Is the same dating as `Weekly`.

See Also

See also the standard dating `Weekly`

Index