The fSeries Package

October 8, 2007

Version 260.72

Date 1997 - 2007

Title Rmetrics - Financial Time Series Objects

Author Diethelm Wuertz and many others, see the SOURCE file

Depends R (>= 2.4.0), robustbase, fCalendar

Maintainer Diethelm Wuertz and Rmetrics Core Team <Rmetrics-core@r-project.org>

Description Environment for teaching “Financial Engineering and Computational Finance”

NOTE SEVERAL PARTS ARE STILL PRELIMINARY AND MAY BE CHANGED IN THE FUTURE. THIS TYPICALLY INCLUDES FUNCTION AND ARGUMENT NAMES, AS WELL AS DEFAULTS FOR ARGUMENTS AND RETURN VALUES.

LazyLoad yes

LazyData yes

License GPL Version 2 or later

URL http://www.rmetrics.org

R topics documented:

aggregate ..................................................... 2
apply .......................................................... 3
as ............................................................. 4
attach ......................................................... 6
bind ............................................................ 7
colCum ........................................................ 8
cor ............................................................ 9
SpecialDailySeries .......................................... 10
dim ........................................................... 13
durations ..................................................... 14
lag ............................................................ 15
aggregate

timeSeries Class, Functions and Methods

Description
Computes Summary Statistics of Data Subsets

Usage

## S3 method for class 'timeSeries':
aggregate(x, by = c("monthly", "quarterly"),
          FUN = colMeans, units = NULL, ...)

Arguments

by [aggregate] -
a character string denoting the aggregation period, either "monthly" or "quarterly".
FUN the function to be applied.
units an optional character string, which allows to overwrite the current column names of a timeSeries object. By default NULL which means that the column names are selected automatically.
x an object of class timeSeries.
... arguments passed to other methods.

Value
returns an aggregated S4 object of class timeSeries.
Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

## data -
   x = as.timeSeries(data(msft.dat))

## aggregate -
   aggregate(x)

apply

Applies Functions Over timeSeries Margins

Description

Applies Functions Over timeSeries Margins

Usage

fapply(x, from, to, FUN, ...)

Arguments

from, to  starting date and end date, to must be after from.
FUN      the function to be applied.
x        an object of class timeSeries.
...      arguments passed to other methods.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

## data -

as

timeSeries Class, Coercion and Transformation

Description

A collection and description of functions and methods dealing with the coercion of 'timeSeries' objects.

Functions to create 'timeSeries' objects from other objects:
as.timeSeries
as.timeSeries.default
as.timeSeries.numeric
as.timeSeries.data.frame
as.timeSeries.matrix
as.timeSeries.ts
as.timeSeries.character
as.timeSeries.zoo

as.vector.timeSeries
as.matrix.timeSeries
as.data.frame.timeSeries
as.ts.timeSeries

Functions to transform 'timeSeries' objects into other objects:

is.timeSeries(object)
## S3 method for class 'numeric':
as.timeSeries(x, ...)
## S3 method for class 'data.frame':
as.timeSeries(x, ...)
## S3 method for class 'matrix':
as.timeSeries(x, ...)
## S3 method for class 'ts':
as.timeSeries(x, ...)
## S3 method for class 'character':
as.timeSeries(x, ...)
## S3 method for class 'zoo':
as.timeSeries(x, ...)

## S3 method for class 'timeSeries':
as.vector(x, mode = "any")  # mode is a character string giving an atomic mode or "list", or (not for 'vector') "any".
## S3 method for class 'timeSeries':
as.matrix(x, ...)
## S3 method for class 'timeSeries':
as.data.frame(x, row.names = NULL, optional = NULL, ...)
## S3 method for class 'timeSeries':
as.ts(x, ...)

Arguments

mode a character string giving an atomic mode or "list", or (not for 'vector') "any".
object an object of class timeSeries.
optional A logical value. If TRUE, setting row names and converting column names (to syntactic names) is optional.
row.names NULL or a character vector giving the row names for the data frame. Missing values are not allowed.

x an object which is coerced according to the generic function.

... arguments passed to other methods.

Value

is.timeSeries returns TRUE or FALSE depending on whether its argument is of timeSeries type or not.

as.timeSeries returns a S4 object of class timeSeries.

as.vector
as.data.frame
as.matrix
as.ts
return depending on the generic function a numeric vector, a data frame, a matrix, or an object of class ts.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

## data - timeSeries:
# Create an artificial timeSeries object:
myFinCenter <<- "GMT"
charvec = timeCalendar()
data = matrix(rnorm(12))
TS = timeSeries(data, charvec, units = "RAND")
TS

## Test for timeSeries:
is.timeSeries(TS)

## As Vector:
as.vector(TS)

## As Matrix or Data Frame:
as.matrix(TS)
as.data.frame(TS)

## As Univariate Object of Class 'ts':
as.ts(TS)
attach

Attach a timeSeries to the search path

Description

A collection and description of functions and methods dealing with the attachment of timeSeries objects to the search path.

attach attaches a 'timeSeries' object,
detach detaches a 'timeSeries' object [see base package].

Usage

## S3 method for class 'timeSeries':
attach(what, pos = 2, name = deparse(substitute(what)),
warn.conflicts = TRUE)

Arguments

name [attach] -
alternative way to specify the database to be attached. See for details help(attach,package=base)
pos [attach] -
an integer specifying position in search() where to attach the database. See for details help(attach,package=base).
warn.conflicts [attach] -
a logical value. If TRUE, warnings are printed about conflicts from attaching the database, unless that database contains an object .conflicts.OK. A conflict is a function masking a function, or a non-function masking a non-function. See for details help(attach,package=base).
what [attach] -
database to be attached. This may currently be a timeSeries object, a data.frame or a list or a R data file created with save or NULL or an environment. See for details help(attach,package=base).

Note

Preliminary, further work has to be done.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.
Examples

```r
## data -
  x = as.timeSeries(data(msft.dat))[1:10, ]

## attach -
  attach(x)
  High - Low
```

---

**bind**  
*Bind two timeSeries objects*

Description

Binds two timeSeries objects either by column or row.

Usage

```r
## S3 method for class 'timeSeries':
  cbind(x, y, units = NULL)
## S3 method for class 'timeSeries':
  rbind(x, y, units = NULL)
```

Arguments

- `units` an optional character string, which allows to overwrite the current column names of a timeSeries object. By default NULL which means that the column names are selected automatically.
- `x, y` two objects of class timeSeries.

Value

returns a S4 object of class timedate.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## data -
  x = as.timeSeries(data(msft.dat))[1:12, ]

## cbind -
  cbind(x[, "Open"], returnSeries(x[, "Open"]), units = c("Open", "Return"))

## rbind -
  rbind(x[1:3, "Open"], x[10:12, "Open"])
```
Description

Functions to compute cumulative column statistics.

Usage

```r
## Default S3 method:
colCumsums(x, na.rm = FALSE, ...)
## S3 method for class 'timeSeries':
colCumsums(x, na.rm = FALSE, ...)
## S3 method for class 'zoo':
colCumsums(x, na.rm = FALSE, ...)

## Default S3 method:
colCummaxs(x, na.rm = FALSE, ...)
## S3 method for class 'timeSeries':
colCummaxs(x, na.rm = FALSE, ...)
## S3 method for class 'zoo':
colCummaxs(x, na.rm = FALSE, ...)

## Default S3 method:
colCumprods(x, na.rm = FALSE, ...)
## S3 method for class 'timeSeries':
colCumprods(x, na.rm = FALSE, ...)
## S3 method for class 'zoo':
colCumprods(x, na.rm = FALSE, ...)

## Default S3 method:
colCumreturns(x, method = c("geometric", "simple"), na.rm = FALSE, ...)
## S3 method for class 'timeSeries':
colCumreturns(x, method = c("geometric", "simple"), na.rm = FALSE, ...)
## S3 method for class 'zoo':
colCumreturns(x, method = c("geometric", "simple"), na.rm = FALSE, ...)
```

Arguments

- `method` a character string to indicate if geometric (TRUE) or simple (FALSE) returns should be computed.
- `na.rm` a logical. Should missing values be removed?
- `x` a time series, may be an object of class "matrix", "timeSeries", or "zoo".
cor

d...arguments to be passed.

Value

all functions return an S4 object of class timeSeries.

Note

The method cumsum.timeSeries will be no longer supported use instead colCumsums.

The methods for the "zoo" methods are still preliminary and untested. na.rm is not yet supported in all case, please test.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

## Simulated Monthly Return Data:
x = matrix(rnorm(24), ncol = 2)

## colStats -
colCumsums(x)

cor timeSeries Correlations

Description

A collection and description of functions and methods dealing with correlations between 'time-Series' objects.

cov Computes Covariance from a 'timeSeries' object,
cor Computes Correlations from a 'timeSeries' object.

Usage

## S3 method for class 'timeSeries':
cov(x, y = NULL, use = "all.obs",
   method = c("pearson", "kendall", "spearman"))

## S3 method for class 'timeSeries':
cor(x, y = NULL, use = "all.obs",
   method = c("pearson", "kendall", "spearman"))
Arguments

method

a character string indicating which correlation coefficient (or covariance) is to be computed. One of "pearson" (default), "kendall", or "spearman", can be abbreviated.

use

an optional character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings "all.obs", "complete.obs" or "pairwise.complete.obs".

x

an univariate object of class timeSeries.

y

NULL (default) or a timeSeries object with compatible dimensions to x. The default is equivalent to y = x (but more efficient).

Value

returns the covariance or correlation matrix.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## data -
x = as.timeSeries(data(msft.dat))[, 1:4]
x = 100*returnSeries(x)

## cov -
cov(x[, "Open"], x[, "Close")
cov(x)
```

Description

A collection and description of special daily timeSeries functions.

dummyDailySeries     Creates a dummy daily 'timeSeries' object,
alignDailySeries     Aligns a daily 'timeSeries' to new positions,
rollDailySeries      Rolls daily a 'timeSeries' on a given period,
ohlcDailyPlot        Plots open high low close bar chart.

Usage

dummyDailySeries(x = rnorm(365), units = "X", zone = myFinCenter,
                 FinCenter = myFinCenter)
alignDailySeries(x, method = c("before", "after", "interp", "fillNA"),
include.weekends = FALSE, units = NULL, zone = myFinCenter, FinCenter = myFinCenter)
rollDailySeries(x, period = "7d", FUN, ...)
ohlcDailyPlot(x, volume = TRUE, colOrder = c(1:5), units = 1e6, xlab = c("Date", "Date"), ylab = c("Price", "Volume"),
main = c("O-H-L-C", "Volume"), grid.nx = 7, grid.lty = "solid", ...)

Arguments

colOrder [ohlcDailyPlot] -
an integer vector which gives the order of the prices and the volume in the input object. By default the following order of columns from 1 to 5 is assumed: Open, high, low, close, and volume.

FinCenter a character with the the location of the financial center named as "continent/city".

FUN the function to be applied.
[applySeries] -
a function to use for aggregation, by default colAvgs.

grid.lty, grid.nx [ohlcDailyPlot] -
The type of grid line and the number of grid lines used in the plot.

include.weekends [alignDailySeries] -
a logical value. Should weekend dates be included or removed from the series.

main [ohlcDailyPlot] -
a character string to title the price and volume plot.

method [alignDailySeries] -
the method to be used for the alignment. A character string, one of "before", use the data from the row whose position is just before the unmatched position, or "after", use the data from the row whose position is just after the unmatched position, or "linear", interpolate linearly between "before" and "after".

period [rollDailySeries] -
a character string specifying the rolling period composed by the length of the period and its unit, e.g. "7d" represents one week.

units [allignDailySeries] -
an optional character string, which allows to overwrite the current column names of a timeSeries object. By default NULL which means that the column names are selected automatically.
[ohlcDailyPlot] -
a numeric value, specifying in which multiples the volume should be referenced on the plot labels. By default 1e6, i.e. in units of 1 Million.

volume [ohlcDailyPlot] -
a logilical value. Should a volume plot added to the OHLN Plot. By default TRUE.

x an object of class timeSeries.
xlab, ylab  [ohlcDailyPlot] -
two string vectors to name the x and y axis of the price and volume plot.
zone
the time zone or financial center where the data were recorded.
...
arguments passed to other methods.

Value
dummyDailySeries
creates from a numeric matrix with daily records of unknown dates a timeSeries object with
dummy daily dates.

alignDailySeries
returns from a daily time series with missing holidays a weekly aligned daily timeSeries object

rollDailySeries
returns an object of class timeSeries with rolling values, computed from the function FUN".

ohlcDailyPlot displays a Open-High-Low-Close Plot of daily data records.

Author(s)
Diethelm Wuertz for the Rmetrics R-port.

Examples

## data - Data Frame:
# To work with daily data, the best choice is "GMT"
myFinCenter <<- "GMT"
MSFT = as.timeSeries(data(msft.dat))
head(MSFT)

## Align Daily Series -
# Cut out April Data from 2001:
Close = MSFT[, "Close"]
tsApril01 = window(Close, "2001-04-01", "2001-04-30")
tsApril01
# Align with NA:
tsRet = returnSeries(tsApril01, trim = TRUE)
GoodFriday(2001)
EasterMonday(2001)
alignDailySeries(tsRet, method = "fillNA", include.weekends = FALSE)
alignDailySeries(tsRet, method = "fillNA", include.weekends = TRUE)
# Interpolate:
alignDailySeries(tsRet, method = "interp", include.weekend = FALSE)
alignDailySeries(tsRet, method = "interp", include.weekend = TRUE)

## ohlcDailyPlot -
# ohlcDailyPlot(Close)
Description

A collection and description of functions and methods dealing with columns and rows of 'timeSeries' objects.

- `dim` (Returns the dimension of a 'timeSeries' object,
- `dimnames` (Returns the dimension names of a 'timeSeries' object,
- `colnames<-` (Assigns column names to a 'timeSeries' object,
- `rownames<-` (Assigns row names to a 'timeSeries' object,
- `is.array` (Allows that NCOL and NROW work properly.

Usage

```r
## S3 method for class 'timeSeries':
dim(x)
## S3 method for class 'timeSeries':
dimnames(x)

colnames<-.timeSeries(x) <- value
rownames<-.timeSeries(x) <- value

## S3 method for class 'timeSeries':
is.array(x)
```

Arguments

- `value` a valid value for column names component of `dimnames(x)`. For a "timeSeries" object this is either NULL or a character vector of length the column dimension. Not, row names cannot be assigne for a "timeSeries" object, the function `rownames()` will stop and return an error message.

- `x` an object of class `timeSeries`.

Value

NA

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## NYI
```
durations Computes durations from a timeSeries

Description
Computes durations from an object of class 'timeSeries'.

Usage
durations(x, trim = FALSE, units = c("secs", "mins", "hours"))
durationSeries(...)

Arguments
trim a logical value. By default TRUE, the first missing observation in the return series will be removed.
units [durationSeries] - a character value or vector which allows to set the units in which the durations are measured. By default durations are measured in seconds.
x an object of class timeSeries.
... arguments to be passed.

Value
returns an object of class timeSeries.

Author(s)
Diethelm Wuertz for the Rmetrics R-port.

Examples
## data -
# Microsoft Data:
myFinCenter <<- "GMT"
MSFT = as.timeSeries(data(msft.dat))[1:20, "Open"]
head(MSFT)

## durations -
# Durations in hours: Continuous Returns:
durations(MSFT, units = "hours")
Description
Computes a lagged version of a timeSeries object.

Usage
## S3 method for class 'timeSeries':
lag(x, k = 1, trim = FALSE, units = NULL, ...)

Arguments

k [lagSeries] -
an integer value. The number of lags (in units of observations). By default 1.

trim a logical value. By default TRUE, the first missing observation in the return
series will be removed.

units an optional character string, which allows to overwrite the current column names
of a timeSeries object. By default NULL which means that the column
names are selected automatically.

x an object of class timeSeries.

... arguments passed to other methods.

Value
returns a lagged S4 object of class timeSeries.

Author(s)
Diethelm Wuertz for the Rmetrics R-port.

Examples
## data -
  x = as.timeSeries(data(msft.dat))[1:20, "Open"]

## lag -
# Lag the timeSeries Object:
  lag(x, k = -1:1)
Mathematical timeSeries Operations

Description

A collection and description of functions and methods dealing with mathematical timeSeries operations.

Ops.timeSeries  S3: Arith method for a 'timeSeries' object,
abs  Returns absolute values of a 'timeSeries' object,
sqrt  Returns square root of a 'timeSeries' object,
exp  Returns the exponential values of a 'timeSeries' object,
log  Returns the logarithm of a 'timeSeries' object,
sign  Returns the signs of a 'timeSeries' object,
diff  Differences a 'timeSeries' object,
scale  Centers and/or scales a 'timeSeries' object,
quantile  Returns quantiles of an univariate 'timeSeries'.

Usage

## S3 method for class 'timeSeries':
Ops(e1, e2)
## S3 method for class 'timeSeries':
abs(x)
## S3 method for class 'timeSeries':
sqrt(x)
## S3 method for class 'timeSeries':
exp(x)
## S3 method for class 'timeSeries':
log(x, base = exp(1))
## S3 method for class 'timeSeries':
sign(x)
## S3 method for class 'timeSeries':
diff(x, lag = 1, diff = 1, trim = FALSE, pad = NA, ...)
## S3 method for class 'timeSeries':
scale(x, center = TRUE, scale = TRUE)
## S3 method for class 'timeSeries':
quantile(x, ...)

Arguments

- base  [log] - a positive number. The base with respect to which logarithms are computed. Defaults to e=exp(1).
- center, scale  [scale] -
either a logical value or a numeric vector of length equal to the number of
columns of \( x \).

diff
an integer indicating the order of the difference. By default 1.

e1, e2
[Ops] -
two objects of class \texttt{timeSeries}.

lag
an integer indicating which lag to use. By default 1.

pad
[diffSeries] -
which value should get the padded values? By default \texttt{NA}. Another choice often
used would be zero.

trim
a logical value. By default \texttt{TRUE}, the first missing observation in the return
series will be removed.

x
an object of class \texttt{timeSeries}.

... arguments to be passed.

Value
returns the value from a mathematical or logical operation operating on objects of class \texttt{timeSeries},
or the value computed by a mathematical function.

Author(s)
Diethelm Wuertz for the Rmetrics \texttt{R}-port.

Examples

```r
## data -
# Create an artificial timeSeries object:
myFinCenter <- "GMT"
charvec = timeCalendar()
set.seed(4711)
data = matrix(exp(cumsum(rnorm(12, sd = 0.1))))
TS = timeSeries(data, charvec, units = "TS")
TS

## Ops | +/- * ^ ...
# Mathematical Operations:
TS^2
TS[2:4]
OR = returnSeries(TS)
OR
OR > 0
```
merge

Merge two timeSeries objects

Description

Merges two timeSeries objects.

Usage

```r
## S3 method for class 'timeSeries':
merge(x, y, units = NULL, ...)
```

Arguments

- `units`: an optional character string, which allows to overwrite the current column names of a `timeSeries` object. By default `NULL` which means that the column names are selected automatically.
- `x, y`: two objects of class `timeSeries`.
- `...`: arguments passed to other methods.

Value

returns an S4 object of class `timeSeries`.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## data -
x = as.timeSeries(data(msft.dat))[1:20, "Open"]

## aggregate -
merge(x, returnSeries(x))
```

model.fram

Model Frames for timeSeries Objects

Description

Allows to work with model frames for 'timeSeries' objects.

Usage

```r
## S3 method for class 'timeSeries':
model.frame(formula, data, ...)
```
Arguments

formula  a model formula object.
data    an object of class `timeSeries`.
...    arguments passed to the function `stats::model.frame`.

Details

The function `model.frame` is a generic function which returns in the R-stts framework by default a data.frame with the variables needed to use formula and any ... arguments. In contrast to this the method returns an object of class `timeSeries` when the argument data was not a `data.frame` but also an object of class `timeSeries`.

Value

an object of class `timeSeries`.

Note

This function is preliminary and untested.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

See Also

`model.frame`.

Examples

```r
## data -
# Microsoft Data:
myFinCenter <- "GMT"
MSFT = as.timeSeries(data(msft.dat))[1:12, ]

## model.frame -
# Extract High's and Low's:
model.frame(~ High + Low, data = MSFT)
# Extract Open Prices and their log10's:
base = 10
Open = model.frame(Open ~ log(Open, base = `base`), data = MSFT)
colnames(Open) <- c("MSFT", "log10(MSFT)")
Open
```
A collection and description of functions and methods dealing with special monthly 'timeSeries' objects.

- **countMonthlyRecords** Returns a series with monthly counts of records,
- **isMonthly** Decides if the series consists of monthly records,
- **rollMonthlyWindows** Returns start and end dates for rolling time windows,
- **rollMonthlySeries** Rolls monthly a 'timeSeries' on a given period.

### Usage

- `countMonthlyRecords(x)`
- `isMonthly(x)`
- `rollMonthlyWindows(x, period = "12m", by = "1m")`
- `rollMonthlySeries(x, period = "12m", by = "1m", FUN, ...)`

### Arguments

- **by**
  - a character string specifying the rolling shift composed by the length of the shift and its unit, e.g. "3m" represents quarterly shifts.
- **FUN**
  - the function to be applied.
  - [applySeries](#) - a function to use for aggregation, by default `colAvgs`.
- **period**
  - a character string specifying the rolling period composed by the length of the period and its unit, e.g. "12m" represents one year.
- **x**
  - an object of class `timeSeries`.
- **...**
  - arguments passed to other methods.

### Value

- NA

### Author(s)

Diethelm Wuertz for the Rmetrics R-port.
Examples

```r
## data -
# Microsoft Daily Data Set:
x = as.timeSeries(data(msft.dat))
countMonthlyRecords(x)
isMonthly(x)

## data -
# EDHEC Hedge Funds Monthly Data Set
x = as.timeSeries(data(edhec))
isMonthly(x)
```

---

**Handling Missing Values**

Description

A collection and description of functions for handling missing values in 'timeSeries' objects or in objects which can be transformed into a vector or a two dimensional matrix.

The functions are listed by topic.

- **na.omit** Handles NAs,
- **removeNA** Removes NAs from a matrix object,
- **substituteNA** substitute NAs by zero, the column mean or median,
- **interpNA** interpolates NAs using R’s "approx" function.

Usage

```r
## S3 method for class 'timeSeries':
na.omit(object, method = c("r", "s", "z", "iz", "ie"),
        interp = c("before", "linear", "after"), ...)
removeNA(x, ...)
substituteNA(x, type = c("zeros", "mean", "median"), ...)
interpNA(x, method = c("linear", "before", "after"), ...)
```

Arguments

- **interp, type** [na.omit][substituteNA] -
  Three alternative methods are provided to remove NAs from the data: type="zeros" replaces the missing values by zeros, type="mean" replaces the missing values by the column mean, type="median" replaces the missing values by the column median.

- **method** [na.omit] -
  Specifies the method how to handle NAs. One of the applied vector strings:
method="s" na.rm = FALSE, i.e. do nothing, method="r" remove NAs, method="z" substitute NAs by zeros, method="ir" interpolate NAs and remove NAs at the beginning and end of the series, method="iz" interpolate NAs and substitute NAs at the beginning and end of the series, method="ie" interpolate NAs and extrapolate NAs at the beginning and end of the series, [interpNA] -

Specifies the method how to interpolate the matrix column by column. One of the applied vector strings: method="linear", method="before" or method=after".

For the interpolation the function approx is used.

object
an object of class("timeSeries").
x
a numeric matrix, or any other object which can be transformed into a matrix through x = as.matrix(x, ...). If x is a vector, it will be transformed into a one-dimensional matrix.

... arguments to be passed to the function as.matrix.

Details

Missing Values in Price and Index Series:

Applied to timeSeries objects the function removeNA just removes rows with NAs from the series. For an interpolation of time series points one can use the function interpNA. Three different methods of interpolation are offered: "linear" does a linear interpolation, "before" uses the previous value, and "after" uses the following value. Note, that the interpolation is done on the index scale and not on the time scale.

Missing Values in Return Series:

For return series the function substituteNA may be useful. The function allows to fill missing values either by method="zeros", the method="mean" or the method="median" value of the appropriate columns.

Note

The functions removeNA, substituteNA and interpNA are older implementations. Please use in all cases if possible the new function na.omit.

Author(s)

Raphael Gottardo for the knn function,
Diethelm Wuertz for the Rmetrics R-port.

References

## Examples

### Create a Matrix with NAs:

```r
X = matrix(rnorm(100), ncol = 5)
# a single NA inside:
X[3, 5] = NA
# three in a row inside:
X[17, 2:4] = c(NA, NA, NA)
# three in a column inside:
X[13:15, 4] = c(NA, NA, NA)
# two at the right border:
X[11:12, 5] = c(NA, NA)
# one in the lower left corner:
X[20, 1] = NA
print(X)
```

### removeNA -

- Remove rows with NA's

```r
removeNA(X)
# Now we have only 12 lines!
```

### substituteNA -

- Substitute NA's by zeros or column mean

```r
substituteNA(X, type = "zeros")
substituteNA(X, type = "mean")
```

### interpNA -

- Interpolate NA's linearly:

```r
interpNA(X, method = "linear")
# Note the corner missing value cannot be interpolated!
# Take previous values in a column:
interpNA(X, method = "before")
# Also here, the corner value is excluded
```

### orderColnames

Reorders column names of a time series

Description

A collection and description of functions and methods dealing with the rearrangement of column names of ‘timeSeries’ objects.

- `orderColnames` Returns ordered column names of a time Series,
- `sortColnames` Returns sorted column names of a time Series,
- `sampleColnames` Returns sampled column names of a time Series,
- `statsColnames` Returns statistically rearranged column names,
- `pcaColnames` Returns PCA correlation ordered column names,
- `hclustColnames` Returns hierarchical clustered column names.
Usage

orderColnames(x, ...)  
sortColnames(x, ...)  
sampleColnames(x, ...)  
statsColnames(x, FUN = colMeans, ...)  
pcaColnames(x, robust = FALSE, ...)  
hclustColnames(x, method = c("euclidean", "complete"), ...)  

Arguments

FUN  
a character string indicating which statistical function should be applied. By default statistical ordering operates on the column means of the time series.

method  
a character string with two elements. The first determines the choice of the distance measure, see dist, and the second determines the choice of the agglomeration method, see hclust.

robust  
a logical flag which indicates if robust correlations should be used.

x  
an object of class timesSeries or any other rectangular object which can be transformed by the function as.matrix into a numeric matrix.

...  
further arguments to be passed, see details.

Details

Statistically Motivated Rearrangement

The function statsColnames rearranges the column names according to a statical measure. These measure must operate on the columns of the time series and return a vector of values which can be sorted. Typical functions are those listed in in help page colStats but one can also create his own functions which compute for example risk or any other statistical measure. The ... argument allows to pass additional arguments to the underlying function FUN.

PCA Ordering of the Correlation Matrix

The function pcaColnames rearranges the column names according to the PCA ordered correlation matrix. The argument robust allows to select between the use of the standard cor and computation of robust correlations using the function covMcd from contributed R package robustbase. The ... argument allows to pass additional arguments to the two underlying functions cor or covMcd. E.g. adding method="kendall" to the argument list calculates Kendall’s rank correlations instead the default which calculates Person’s correlations.

Ordering by Hierarchical Clustering

The function pcaColnames uses the hierarchical clustering approach hclust to rearrange the column names of the time series.

Value

returns a vector of character string, the rearranged column names.
Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## data -
edhec = as.timeSeries(data(edhec))
colnames(edhec) = abbreviate(colnames(edhec), 6)

## sortColnames -
# Sort alphabetically
sortColnames(edhec)

## hclustColnames -
head(edhec[, hclustColnames(edhec)])
```

Description

Functions to plot timeSeries objects.

Usage

```r
## S3 method for class 'timeSeries':
plot(x, ...)

## S3 method for class 'timeSeries':
lines(x, ...)

## S3 method for class 'timeSeries':
points(x, ...)
```

Arguments

- `x` an object of class `timeSeries`
- `...` arguments passed to other methods.

Value

a plot or plot elements of an object of class `timeSeries`.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.
Examples

```r
## data -
myFinCenter <<- "GMT"
EDHEC = as.timeSeries(data(edhec))[1:12, 1:4]
colnames(EDHEC) <- abbreviate(colnames(EDHEC), 6)

## plot -
plot(EDHEC[,1], type = "o", col = "steelblue",
    main = "EDHEC", xlab = "1997", ylab = "Return")
```

print

Prints timeSeries objects

Description
Prints objects of class timeSeries.

Usage

```r
show.timeSeries(object)
```

Arguments

```r
object an object of class timeSeries.
```

Value

prints an object of class timeSeries.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## print -
myFinCenter <<- "GMT"
EDHEC = as.timeSeries(data(edhec))[1:12, 1:4]
colnames(EDHEC) <- abbreviate(colnames(EDHEC), 6)
print(EDHEC)
```
returns  Calculations of Financial Returns

Description

Functions to calculate financial returns.

Usage

```r
returns(x, ...)  
```

## Default S3 method:
```r
returns(x, method = c("continuous", "discrete", "compound", "simple"),  
    percentage = FALSE, ...)  
```

## S3 method for class 'timeSeries':
```r
returns(x, method = c("continuous", "discrete", "compound", "simple"),  
    percentage = FALSE, na.rm = TRUE, trim = TRUE, ...)  
```

## S3 method for class 'zoo':
```r
returns(x, method = c("continuous", "discrete", "compound", "simple"),  
    percentage = FALSE, na.rm = TRUE, trim = TRUE, ...)  
```

getReturns(...)
returnSeries(...)  

Arguments

- `percentage`  
  a logical value. By default FALSE, if TRUE the series will be expressed in percentage changes.
- `method`  
- `na.rm`  
- `trim`  
- `x`  
  an object of class `timeSeries`.
- `...`  
  arguments to be passed.

Value

all functions return an object of class `timeSeries`.

Note

The functions `returnSeries`, `getReturns`, are synonyms for `returns.timeSeries`.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.
## data -

```r
# Microsoft Data:
myFinCenter <- "GMT"
MSFT = as.timeSeries(data(msft.dat))[1:10, 1:4]
head(MSFT)
```

## returnSeries -

```r
# Continuous Returns:
returns(MSFT)
# Discrete Returns:
returns(MSFT, type = "discrete")
# Don't trim:
returns(MSFT, trim = FALSE)
# Use Percentage Values:
returns(MSFT, percentage = TRUE, trim = FALSE)
```

---

### rowCum

**Cumulated Column Statistics**

#### Description

Functions to compute cumulative row Statistics.

#### Usage

```r
## Default S3 method:
rowCumsums(x, na.rm = FALSE, ...)
## S3 method for class 'timeSeries':
rowCumsums(x, na.rm = FALSE, ...)
## S3 method for class 'zoo':
rowCumsums(x, na.rm = FALSE, ...)
```

#### Arguments

- `na.rm` a logical. Should missing values be removed?
- `x` a time series, may be an object of class "matrix", "timeSeries", or "zoo".
- `...` arguments to be passed.

#### Value

all functions return an S4 object of class `timeSeries`.

#### Note

The methods for the "zoo" methods are still preliminary and untested. `na.rm` is not yet supported in all case, please test.
spreads

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## Simulated Monthly Return Data:
x = matrix(rnorm(24), ncol = 2)

## colStats -
rowCumsums(x)
```

spreads

Calculations of spreads and mid quotes

Description

Functions to calculate spreads and midquotes from price streams.

Usage

```r
spreads(x, which = c("Bid", "Ask"), tickSize = NULL)
midquotes(x, which = c("Bid", "Ask"))

midquoteSeries(...)  
spreadSeries(...)
```

Arguments

- **tickSize**
  the default is NULL to simply compute price changes in original price levels. If ticksize is supplied, the price changes will be divided by the value of inTicksOfSize to compute price changes in ticks.

- **which**
  a vector with two character strings naming the column names of the time series from which to compute the mid quotes and spreads. By default these are bid and ask prices with column names c("Bid", "Ask").

- **x**
  an object of class `timeSeries`.

- **...**
  arguments to be passed.

Value

all functions return an object of class `timeSeries`.

Note

The functions `returnSeries`, `getReturns`, `midquoteSeries`, `spreadSeries` are synonyms for `returns`, `midquotes`, and `spreads`. 
Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## data -
# Microsoft Data:
myFinCenter <- "GMT"
MSFT = as.timeSeries(data(msft.dat))[1:10, 1:4]
head(MSFT)
```

## midquotes -

## spreads -

---

TimeSeriesSubsettings

timeSeries Subsetting

Description

A collection and description of functions and methods for subsetting timeSeries objects.

Usage

```r
# S3 method for class 'timeSeries':
x[i = min(1, nrow(x@Data)):nrow(x@Data),
  j = min(1, ncol(x@Data)):ncol(x@Data)]
```

```r
# S3 method for class 'timeSeries':
window(x, from, to, ...)
```

```r
# S3 method for class 'timeSeries':
head(x, n = 6, recordIDs = FALSE, ...)
```

```r
# S3 method for class 'timeSeries':
tail(x, n = 6, recordIDs = FALSE, ...)
```

```r
# S3 method for class 'timeSeries':
outliers(x, sd = 10, complement = TRUE, ...)
```

```r
# S3 method for class 'timeSeries':
```
cut(x, from, to, ...)

Arguments

complement [outlierSeries] - a logical flag, should the outlier series or its complement be returns, by default TRUE which returns the series free of outliers.

from, to starting date and end date, to must be after from.

i, j ["["]- index arguments used for subsettings.

n [head][tail] - an integer specifying the number of lines to be returned. By default n=6.

recordIDs [head][tail] - a logical value. Should the recordIDs returned together with the data matrix and time series positions?

sd [outlierSeries] - a numeric value of standard deviations, e.g. 10 means that values larger or smaller than ten times the standard deviation will be removed from the series.

x an object of class timeSeries.

... arguments passed to other methods.

Value

all functions return an object of class timeSeries.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## data -
# Create an artificial timeSeries object:
myFinCenter <- "GMT"
charvec = timeCalendar()
set.seed(4711)
data = matrix(exp(cumsum(rnorm(12, sd = 0.1))))
tS = timeSeries(data, charvec, units = "tS")
tS

## "[" -
tS[1:3, ]

## head -
head(tS)
```
**summary** |  
---|---

*Object Summary*

**Description**

produces a result summary of a timeSeries object.

**Usage**

```r
## S3 method for class 'timeSeries':
summary(object, ...)
```

**Arguments**

- `object` | an object of class `timeSeries`.
- `...` | arguments passed to other methods.

**Value**

returns a summary report for an object of class `timeSeries`.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

**Examples**

```r
## data -
EDHEC = as.timeSeries(data(edhec))[1:12, 1:4]
colnames(EDHEC) <- abbreviate(colnames(EDHEC), 4)

## summary -
summary(EDHEC)
```

**time** |  
---|---

*timeSeries, Positions*

**Description**

A collection and description of functions and methods extracting and modifying positions on `timeSeries` objects.

The functions and methods for the Generation of `timeSeries` Objects are:

- `seriesPositions` | Extracts positions slot from a `timeSeries`.
- `newPositions<-` | Modifies positions of a `timeSeries` object,
time

time.timeSeries  
Extracts time positions from a 'timeSeries',
sample.timeSeries  
Resamples a 'timeSeries' object in time,
sort.timeSeries  
Sorts reverts a 'timeSeries' object in time,
rev.timeSeries  
Reverts a 'timeSeries' object in time,
start.timeSeries  
Extracts start date of a 'timeSeries' object,
end.timeSeries  
Extracts end date of a 'timeSeries' object.

Usage

seriesPositions(object)
newPositions(object) <- value

## S3 method for class 'timeSeries':
time(x, ...)

## S3 method for class 'timeSeries':
start(x, ...)

## S3 method for class 'timeSeries':
end(x, ...)

## S3 method for class 'timeSeries':
sample(x, ...)

## S3 method for class 'timeSeries':
sort(x, ...)

## S3 method for class 'timeSeries':
rev(x)

Arguments

method [alignDailySeries] - 
the method to be used for the alignment. A character string, one of "before", use the data from the row whose position is just before the unmatched position, or "after", use the data from the row whose position is just after the unmatched position, or "linear", interpolate linearly between "before" and "after".

object [is][seriesData][seriesPositions][summary] - an object of class timeSeries.

value a valid value for that component of newPositions(x), i.e. an object of class "timeDate" with appropriate length.

x [as] -
a matrix type object to be converted. [as.vector][as.matrix][as.data.frame] - [applySeries] - [cut][end][mergeSeries][plot][print][rev][start] - an object of class timeSeries.

... arguments passed to other methods.
Value

timeSeries
read.timeSeries
as.timeSeries
return a S4 object of class timeSeries.

seriesData
seriesPositions
extract the @Data and @position slots from a timeSeries object. Thus, seriesData returns an object of class matrix, and seriesPositions returns an object of class timeDate.

is.timeSeries
returns TRUE or FALSE depending on whether its argument is of timeSeries type or not.

aggregateSeries
applySeries
cutSeries
mergeSeries
returnSeries
revSeries
return a S4 object of class timeSeries.

de, start
return a S4 object of class timedate. These are the start and end dates of a timeSeries object.

as.vector
as.matrix
as.data.frame
these are methods which convert a S4 object of class timeSeries either to a vector, a matrix or to a data frame.

plot
lines
points
print
plot and print methods for an object of class timeSeries. Note that the plot function requires the packages its and Hmisc.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

## Create dummy timeSeries:
X = timeSeries(matrix(rnorm(24), 12), timeCalendar())
TimeSeriesClass

## seriesPositions -
seriesPositions(X)

---

### Description

A collection and description of functions and methods dealing with regular and irregular 'timeSeries' objects. Dates and times are implemented as 'timeDate' objects. Included are functions and methods for the generation and representation of 'timeSeries' objects, and for mathematical operations.

#### Functions to generate and modify 'timeSeries' objects:

- `timeSeries` Creates a 'timeSeries' object from scratch,
- `readSeries` Reads a 'timeSeries' from a spreadsheet file,
- `applySeries` Applies a function to margins of a 'timeSeries',
- `orderStatistics` Computes order statistic of a 'timeSeries'.

#### Data Slot and classification of 'timeSeries' objects:

- `seriesData` Extracts data slot from a 'timeSeries',
- `isUnivariate` Tests if a 'timeSeries' object is univariate,
- `isMultivariate` Tests if a 'timeSeries' object is multivariate.

### Usage

```r
# Generate a timeSeries object
x <- timeSeries(data = 1:10, charvec = c("a", "b", "c"), units = "days")

# Read a timeSeries from a spreadsheet file
y <- readSeries(file = "data.csv", zone = "America/New_York")

# Apply a function to margins of a timeSeries
applySeries(x, FUN = mean)

# Compute order statistic of a timeSeries
orderStatistics(x)

# Extract data slot from a timeSeries
seriesData(x)

# Test if a timeSeries object is univariate
isUnivariate(x)

# Test if a timeSeries object is multivariate
isMultivariate(x)
```

```r
# Example with multiple timeSeries objects
z <- list(timeSeries = c(1, 2, 3), timeDate = c(1, 2, 3))
```

```r
# Example with multiple timeSeries objects and dates
w <- list(timeSeries = c(1, 2, 3), timeDate = c(1, 2, 3), timeSeries = c(4, 5, 6))
```
Arguments

by

[applySeries] -
a character either "monthly" or "quarterly". The default value is "monthly". Only operative when both arguments from and to have their default values NULL. In this case the function FUN will be applied to monthly or quarterly periods.

charvec a character vector of dates and times.

data a data.frame or a matrix object of numeric data.

documentation optional documentation string, or a vector of character strings.

file the filename of a spreadsheet data set from which to import the data records.

FinCenter a character with the the location of the financial center named as "continent/city".

header a logical value indicating whether the file contains the names of the variables as its first line. If missing, the value is determined from the file format: ‘header’ is set to ‘TRUE’ if and only if the first row contains one fewer field than the number of columns.

format the format specification of the input character vector,

[as.timeSeries] -
a character string with the format in POSIX notation to be passed to the time series object.

from, to starting date and end date, to must be after from.

FUN the function to be applied.

[applySeries] -
a function to use for aggregation, by default colAvgs.

object [is][seriesData][seriesPositions][show][summary] - an object of class timeSeries.

recordIDs a data frame which can be used for record identification information.

[print] -
a logical value. Should the recordIDs printed together with the data matrix and time series positions?

sep [readSeries] -
the field seperator used in the spreadsheet file to separate columns.

title an optional title string, if not specified the inputs data name is deparsed.

units [applySeries][lag][returnSeries][mergeSeries] -
an optional character string, which allows to overwrite the current column names of a timeSeries object. By default NULL which means that the column names are selected automatically.

[durationSeries] -
a character value or vector which allows to set the units in which the durations are measured. By default durations are measured in seconds.

x [as] -
a matrix type object to be converted.

[as.vector][as.matrix][as.data.frame] -
[applySeries] -
[cut][end][mergeSeries][plot][print][rev][start] -
an object of class timeSeries.
zone the time zone or financial center where the data were recorded.
...
arguments passed to other methods.

Details

Generation of Time Series Objects:

We have defined a `timeSeries` class which is in many aspects similar to the S-Plus class with
the same name, but has also some important differences. The class has seven Slots, the 'Data' slot
which holds the time series data in matrix form, the 'position' slot which holds the time/date as a
character vector, the 'format' and 'FinCenter' slots which are the same as for the 'timeDate' object,
the 'units' slot which holds the column names of the data matrix, and a 'title' and a 'documentation'
slot which hold descriptive character strings. Date and time is managed in the same way as for
timeDate objects.

Value
timeSeries
readSeries
returnSeries
applySeries
return a S4 object of class `timeSeries`.

orderStatistics
returns ...

seriesData
extracts the @Data slot from a `timeSeries` object. Thus, `seriesData` returns an object of
class `matrix`.

isUnivariate
isMultivariate
returns a logical depending if the test is true or not.

plot
lines
points
print
plot and print methods for an object of class `timeSeries`.

Note

These functions were written for Rmetrics users using R and Rmetrics under Microsoft’s Windows
operating system where time zones, daylight saving times and holiday calendars are insufficiently
supported.
Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## data -
# Microsoft Data:
myFinCenter <<- "GMT"
MSFT = as.timeSeries(data(msft.dat))
head(MSFT)

## timeSeries -
# Create a timeSeries Object - The Direct Way ...
Close = MSFT[, 5]
head(Close)
# From Scratch ...
data = as.matrix(MSFT[, 4])
charvec = rownames(MSFT)
Close = timeSeries(data, charvec, units = "Close")
head(Close)
c(start(Close), end(Close))

## window -
# Cut out April Data from 2001:
tsApril01 = window(Close, "2001-04-01", "2001-04-30")
tsApril01

## returnSeries -
# Compute Returns:
args(returnSeries)
# Continuous Returns:
returnSeries(tsApril01)
# Discrete Returns:
returnSeries(tsApril01, type = "discrete")
# Don't trim:
returnSeries(tsApril01, trim = FALSE)
# Use Percentage Values:
tsRet = returnSeries(tsApril01, percentage = TRUE, trim = FALSE)
tsRet

## applySeries -
# Aggregate weekly:
GoodFriday(2001)
to = timeSequence(from = "2001-04-11", length.out = 3, by = "week")
from = to - 6*24*3600
to
applySeries(tsRet, from, to, FUN = sum)
```
Index

*Topic chron
  aggregate, 2
  apply, 3
  as, 3
  attach, 5
  bind, 6
  cor, 9
  dim, 12
  durations, 13
  lag, 14
  math, 15
  merge, 17
  model.fram, 18
  monthly, 19
  orderColnames, 22
  plot, 24
  print, 25
  returns, 26
  SpecialDailySeries, 10
  spreads, 28
  summary, 31
  time, 31
  TimeSeriesClass, 34
  TimeSeriesSubsettings, 29

*Topic math
  na, 20

*Topic univar
  colCum, 7
  rowCum, 27
  [.timeSeries
    (TimeSeriesSubsettings), 29
  
  abs.timeSeries (math), 15
  aggregate, 2
  alignDailySeries
    (SpecialDailySeries), 10
  apply, 3
  applySeries (TimeSeriesClass), 34
  as, 3
  as.data.frame.timeSeries (as), 3
  as.matrix.timeSeries (as), 3
  as.timeSeries (as), 3
  as.ts.timeSeries (as), 3
  as.vector.timeSeries (as), 3
  attach, 5
  bind, 6
  cbind (bind), 6
  colCum, 7
  colCummaxs (colCum), 7
  colCummins (colCum), 7
  colCumprods (colCum), 7
  colCumreturns (colCum), 7
  colCumsums (colCum), 7
  colnames <- .timeSeries (dim), 12
  cor, 9
  countMonthlyRecords (monthly), 19
  cov (cor), 9
  cumsum.timeSeries (colCum), 7
  cut.timeSeries
    (TimeSeriesSubsettings), 29
  diff.timeSeries (math), 15
  dim, 12
  dimnames.timeSeries (dim), 12
  dummyDailySeries
    (SpecialDailySeries), 10
  durations, 13
  durationSeries (durations), 13
  end.timeSeries (time), 31
  exp.timeSeries (math), 15
  fapply (apply), 3
  getReturns (returns), 26
  hclustColnames (orderColnames), 22
  head.timeSeries
    (TimeSeriesSubsettings), 29
interpNA (na), 20
is.array.timeSeries (dim), 12
is.timeSeries (as), 3
isMonthly (monthly), 19
isMultivariate (TimeSeriesClass),
  34
isUnivariate (TimeSeriesClass), 34
lag, 14
lines.timeSeries (plot), 24
log.timeSeries (math), 15
math, 15
merge, 17
midquotes (spreads), 28
midquoteSeries (spreads), 28
model.fram, 18
model.frame, 18
model.frame (model.fram), 18
monthly, 19

na, 20
na.omit.timeSeries (na), 20
newPositions <- (time), 31

ohlcDailyPlot
  (SpecialDailySeries), 10
Ops.timeSeries (math), 15
orderColnames, 22
orderStatistics
  (TimeSeriesClass), 34
outlier.timeSeries
  (TimeSeriesSubsettings), 29

pcaColnames (orderColnames), 22
plot, 24
points.timeSeries (plot), 24
print, 25
quantile.timeSeries (math), 15
rbind (bind), 6
readSeries (TimeSeriesClass), 34
removeNA (na), 20
returns, 26
returnSeries (returns), 26
rev.timeSeries (time), 31
rollDailySeries
  (SpecialDailySeries), 10
rollMonthlySeries (monthly), 19
rollMonthlyWindows (monthly), 19
rowCum, 27
rowCumsums (rowCum), 27
rownames <- .timeSeries (dim), 12
sample.timeSeries (time), 31
sampleColnames (orderColnames), 22
scale.timeSeries (math), 15
seriesData (TimeSeriesClass), 34
seriesPositions (time), 31
show, timeSeries-method (print), 25
show.timeSeries (print), 25
sign.timeSeries (math), 15
sort.timeSeries (time), 31
sortColnames (orderColnames), 22
SpecialDailySeries, 10
spreads, 28
spreadSeries (spreads), 28
sqrt.timeSeries (math), 15
start.timeSeries (time), 31
statsColnames (orderColnames), 22
substituteNA (na), 20
summary, 31
tail.timeSeries
  (TimeSeriesSubsettings), 29
time, 31
timeSeries (TimeSeriesClass), 34
timeSeries-class
  (TimeSeriesClass), 34
TimeSeriesClass, 34
TimeSeriesSubsettings, 29

window.timeSeries
  (TimeSeriesSubsettings), 29