Package ‘tframePlus’

March 4, 2022

Version 2016.7-1
Title Time Frame Coding Kernel Extensions
Description Extensions and additional 'tframe' utilities.
Depends R (>= 2.5.0), tframe (>= 2015.1-1)
Imports stats, utils, graphics, methods, timeSeries
Suggests tfplot, zoo, xts, tis, WriteXLS(>= 1.7.0)
LazyLoad yes
License GPL-2
Author Paul Gilbert <pgilbert.ttf9z@ncf.ca>
Maintainer Paul Gilbert <pgilbert.ttf9z@ncf.ca>
URL http://tsanalysis.r-forge.r-project.org/
NeedsCompilation no
Repository CRAN
Date/Publication 2016-08-26 20:35:53

R topics documented:

as.quarterly ........................................ 2
changeTSrepresentation ................................ 3
rollAggregate ......................................... 4
tfI ...................................................... 5
tfpersp .................................................. 5
TSwriteXLS ............................................. 6

Index 9
as.quarterly

Aggregate Time Series

Description

Convert series to quarterly or annual.

Usage

as.quarterly(x, FUN=sum, na.rm=FALSE, ...)
as.annually(x, FUN=sum, na.rm=FALSE, ...)
as.weekly(x, FUN=sum, na.rm=FALSE, foldFrom=end(x), periodicity = 7)

Arguments

x

a tframed object. (Only monthly is currently working)

FUN

the function to use for aggregating.

na.rm

Logical indicating if NA should be removed from the beginning and end of a series

foldFrom

a date which is used to determine the end of weeks.

periodicity

the number of periods in a week.

... additional arguments passed to aggregate

Details

Functions as.quarterly and as.annually uses aggregate, but shifts the data to match usual economic and financial quarters or years (whereas aggregate simply groups together a number of periods corresponding to the new frequency starting with the first observation). These functions should give the same result as aggregate if the data starts and ends on quarter (annual) boundaries. If the data does not start and end on quarter (annual) boundaries then NA will be put in where data is incomplete, and the quarter (year) removed if na.rm=TRUE.

Monthly to annual gives the aggregate by converting first to quarterly, which is not exactly correct.

To weekly by periodicity groupings from foldFrom. Partial weeks in the beginning or end are padded with NA and dropped if na.rm=FALSE. (as.weekly has been tested only with daily 7 day weeks.)

Value

time series

See Also

tfwindow aggregate
changeTSrepresentation

Change the class of a (Multivariate) Time Series

Description

Change the class of a (Multivariate) Time Series

Usage

changeTSrepresentation (x, newRepresentation)

Arguments

x a multivariate time series object.
newRepresentation function or character string indicating the function to use to change the time series class.

Details

This function attempts to convert the time series to the new class using newRepresentation.

If the representation is a character string then it is applied using do.call(TSrepresentation, list(x, dates)) where x is the time series (matrix) to be return and dates are determined by as.Date(time(x)) where x is the default representation of the data. If representation is not a character string then it should be a function and is applied using TSrepresentation(x, dates)

Value

a multivariate time series object.
rollAggregate

Rolling Time Series Aggregate

Description

Aggregate time series using a rolling window.

Usage

```r
rollAggregate(x, FUN=sum, na.rm=FALSE, aggPeriods=4, ...)```

Arguments

- `x`: a time series object.
- `FUN`: the function to use for aggregating.
- `na.rm`: Logical indicating if NA should be removed from the beginning and end of a series.
- `aggPeriods`: the number of observations to use in the rolling window.
- `...`: additional arguments passed to `apply`

Details

Functions `rollAggregate` uses `apply` with `FUN` on a rolling window set by `aggPeriods`. Partial windows may result in NA, depending on `FUN`. NA in the beginning or end are dropped if `na.rm=TRUE`. The frequency of the resulting series remains the same as the original, but the result will be `aggPeriods` shorter (i.e. have NAs). The default will be a four period moving sum (which roughly converts weekly data to something approximating monthly, but at a weekly frequency).

Value

time series

See Also

`as.annually, as.quarterly, as.weekly, apply`

Examples

```r
rollAggregate(ts(1:20, start = c(1999,1), frequency=1))
rollAggregate(ts(1:20, start = c(1999,1), frequency=12), aggPeriods=3)

if(require("zoo")) {
  z <- zoo(rnorm(100), order.by = Sys.Date() + 1:100)
  rollAggregate(as.weekly(z), aggPeriods=4, FUN=mean)
  require("tfplot")
  tfplot(as.weekly(z),rollAggregate(as.weekly(z), aggPeriods=4, FUN=mean, na.rm=TRUE))
  tfplot(z,rollAggregate(z, aggPeriods=28, FUN=mean))
}
**tfI**

*Integrate Multivariate Time Series*

**Description**

Integrate multivariate time series using `cumsum`

**Usage**

```r
tfI(x, I=rep(TRUE, NCOL(x)), t0=rep(0, NCOL(x)))
```

**Arguments**

- **x**: a multivariate time series object.
- **I**: a logical of length equal the number of series in `x` indicating which series to integrate, default all.
- **t0**: a vector of length equal the number of series in `x` indicating initial (time zero) values, defaults to zero.

**Details**

This function first prepends `t0` to `x` and then applies `cumsum` to each series indicated by `I`. The first value is then dropped from the result.

**Value**

a multivariate time series object of the same dimension as `x`.

**See Also**

`tfL` `tfwindow` `aggregate` `cumsum` `as.quarterly`

---

**tfpersp**

*Plot Series With `persp`*

**Description**

Plot series with `persp`

**Usage**

```r
tfpersp(x, tf=tfspan(x), start=tfstart(tf), end=tfend(tf),
theta = -30, phi = 15, scale = FALSE,
xlab = "Time", ylab = "", zlab = ",
aspect= c(0.5, 0.5), #y/time, z/time,
ticktype="detailed", ltheta = -120, lphi = 15,
...)```

Arguments

- **x**: multivariate time series (a tframed object).
- **tf**: a tframe to set the start and end of the plot.
- **start**: start period for the plot.
- **end**: end period for the plot.
- **theta**: argument passed to `persp`
- **phi**: argument passed to `persp`
- **scale**: argument passed to `persp`
- **xlab**: argument passed to `persp`
- **ylab**: argument passed to `persp`
- **zlab**: argument passed to `persp`
- **aspect**: argument passed to `persp`
- **ticktype**: argument passed to `persp`
- **ltheta**: argument passed to `persp`
- **lphi**: argument passed to `persp`
- **...**: additional arguments passed to `persp`

Details

This function uses passes a multivariate time series to `persp`.

Value

Depends

See Also

tfwindow, `persp`

---

**TSwriteXLS**  
*Write Time Series to an .xls Spreadsheet File*

**Description**

Write one or more time series to a .xls spreadsheet file

**Usage**

```r
TSwriteXLS(x, ..., FileName="R.xls", SheetNames=NULL, 
            dateHeader="date", verbose = FALSE)
TSwriteCSV(x, FileName="R.csv", dateFormat=1, dateHeader="date")
```
Arguments

- **x**: a time series or matrix of time series.
- **...**: additional time series objects (like x).
- **FileName**: a string to use for the file name.
- **SheetNames**: logical indicating if NA should be removed from the beginning and end of a series.
- **dateFormat**: an integer indicating the format to use for dates. (See details.)
- **dateHeader**: string or vector of strings to use as header on date column.
- **verbose**: logical passed to WriteXLS.

Details

This function uses **WriteXLS** to write time series data to an .xls spreadsheet file. If the object `x` has multiple series then they will be put on the same sheet. Additional objects in ... will be put on additional pages.

The first line of each sheet will have column headers using `seriesNames` from the series in the column. The first column will have dates and the heading for that column will be determined by `dateHeader`. This can be a vector of length equal the number of sheets, of a single character string, in which case it is replicated for each sheet. The `dateHeader` is a might be useful for indication information like the date of the last observation in a known place on the sheet (A1) for indicating on a graph. For convenience, the year, period, and a character representation of the period are indicated in columns 2 to 4. This is followed by columns of the data.

If `SheetNames` is `NULL` (the default) then the sheet names will be generated from the first series name in each object. If `SheetNames` is supplied then it should have an element for `x` and for each object in ... .

If **WriteXLS** does not find a version of perl with appropriate modules then a work around is attempted using **save**.

**WriteCSV** handles only a single object `x` which can have multiple series. These are written as columns in the comma separated value file. `dateFormat` can be 0 indicating no date, 1 indicating a format like "Jan 1969", 2 indicating a format like 1969,1, or 3 indicating a format like 1969,"Jan". Only 0 or 1 are supported for data other than monthly and quarterly frequencies.

Value

logical

See Also

**WriteXLS**

Examples

```r
tofile <- tempfile()
z <- ts(1:10, start=c(1999,2), freq=12)
seriesNames(z) <- "ser 1"
```

```r
```
TSwriteXLS(z, FileName=tofile)

zz <- tbind(z, diff(z))
seriesNames(zz) <- c("ser 1", "diff")
TSwriteXLS(zz, FileName=tofile, SheetNames="2 series")

zz <- ts(1:10, start=c(1999,1), freq=1)
seriesNames(zz) <- "annual ser"
TSwriteXLS(z, zz, FileName=tofile, SheetNames=c("monthly", "annual"))

unlink(tofile)
Index

* ts
  as.quarterly, 2
  changeTSrepresentation, 3
  rollAggregate, 4
  tfI, 5
  tfpersp, 5
  TSwriteXLS, 6

* utilities
  as.quarterly, 2
  changeTSrepresentation, 3
  rollAggregate, 4
  tfI, 5
  tfpersp, 5
  TSwriteXLS, 6

aggregate, 2, 5
apply, 4
as.annually, 4
as.annually (as.quarterly), 2
as.quarterly, 2, 4, 5
as.weekly, 4
as.weekly (as.quarterly), 2
changeTSrepresentation, 3
cumsum, 5
persp, 6
rollAggregate, 4
tfI, 5
tfl, 5
tfpersp, 5
tfwindow, 2, 5, 6
TSwriteCSV (TSwriteXLS), 6
TSwriteXLS, 6
WriteXLS, 7