Package ‘fts’

September 14, 2018

Title R Interface to 'tslib' (a Time Series Library in C++)

Version 0.9.9.2

Maintainer Whit Armstrong <armstrong.whit@gmail.com>

Author Whit Armstrong <armstrong.whit@gmail.com>

Depends utils, stats, zoo

LinkingTo BH

Description Fast operations for time series objects.

License GPL-3

BugReports http://github.com/armstrtw/fts/issues

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-09-14 07:36:49 UTC

R topics documented:

apply *Apply Function*

Description

Apply a function to the rows or columns of an fts object

Usage

```
column.apply(x, FUN, ...)
row.apply(x, FUN, ...)
```

Arguments

x	An Fts object
FUN	function to be applied
...	further arguments to function

Value

an Fts object or vector depending on the fuction type

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="months",length.out=24),data=1:24)
y <- fts(index=seq(from=Sys.Date(),by="months",length.out=24),data=1:24)

z <- cbind(x,y)

## returns vector
z.col.sum <- column.apply(z,sum)

## returns fts
z.row.sum <- row.apply(z,sum)
```

`as.fts` *Convert from/to fts*

Description

convert an object into an fts and vice versa

Usage

```
as.fts(x)
```

Arguments

`x`	an R matrix or data.frame

Details

converts a rectanular object into an Fts object must be able to convert rownames into some form of dates

Value

an Fts object

Author(s)

Whit Armstrong

Examples

```
N <- 100
xm <- matrix(rnorm(N))
dts <- format(seq(from=Sys.Date(),length.out=N,by="days"),"%Y-%m-%d")
rownames(xm) <- dts
x.from.m <- as.fts(xm)
x.from.df <- as.fts(data.frame(asofdate=dts,my.data=xm))
```

`event.dates` *Extract Dates*

Description

Extract the dates from a one column LOGICAL Fts object where value is TRUE

Usage

```
event.dates(x)
```

Arguments

`x`	An Fts object

Details

removes NA values before extracting dates

Value

a vector of dates

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),data=rnorm(100))
x.bool <- x > 10
event.dates(x.bool)

## ignores NA's
x.bool[10:20] <- NA
event.dates(x.bool)
```

expanding — *Expanding Window Functions*

Description

apply summary functions on an expanding basis

Usage

```
expanding.max(x)
expanding.min(x)
```

Arguments

x	An Fts object

Details

apply a function that takes a vector and returns a scalar on an expanding basis to an fts object

Value

an fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),data=rnorm(100))

x.emax <- expanding.max(x)
x.emin <- expanding.min(x)
```

fill — *Fill Missing Values*

Description

Fill a missing value (NA) with any of previous value, next value, or a user supplied value.

Usage

```
fill.fwd(x)
fill.bwd(x)
fill.value(x,value)
```

Arguments

x	An Fts object
value	a value to replace the missing values

Value

an Fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=50),rnorm(50))
x[x > 0,] <- NA
fill.fwd(x)

x <- fts(index=seq(from=Sys.Date(),by="days",length.out=50),rnorm(50))
x[x > 0,] <- NA
fill.bwd(x)

x <- fts(index=seq(from=Sys.Date(),by="days",length.out=50),rnorm(50))
x[x > 0,] <- NA
fill.value(x,100.0)
```

`frequency.convert` *Change Frequencies*

Description

convert a time series from a higher frequency to a lower frequency or from an irregular frequency to a regular frequency

Usage

```
to.weekly(x)
to.monthly(x)
to.quarterly(x)
to.day.of.week(x,day.of.week,beginning.of.period=TRUE)
```

Arguments

x	An Fts object.
day.of.week	a numerical value indicating the day of week following POSIXlt conventions.
beginning.of.period	whether to shift the sampling dates to the beginning of period dates.

Value

an Fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=500),data=1:500)

to.weekly(x)
to.monthly(x)
to.quarterly(x)
```

`fts` *Fts: a fast timeseries library*

Description

create an fts object by specifying index and data

Usage

```
fts(index,data)
```

Arguments

`index`	a vector of dates
`data`	a matrix, dataframe, or vector

Details

fts is an S3 class in which the fts object is represented as a native R matrix and the dates are attached as an attribute to the matrix

Value

a fts object

Author(s)

Whit Armstrong

See Also

`as.fts`

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="months",length.out=24),data=1:24)
y <- fts(index=seq(from=Sys.Date(),by="months",length.out=12),data=13:24)
xx <- x[1:10,]

## intersection of dates is taken for Arith methods
xyp <- x + y
xys <- x - y
xym <- x * y
xyd <- x / y
xyg <- x > y
xyl <- x < y

cxy <- cbind(x,y)
rxy <- rbind(x,y)
print(x)
plot(x)
```

---

fts.logical — *Logical subsets of objects*

Description

Find subsets of logical objects

Usage

```
col.any(x)
col.all(x)
row.any(x)
row.all(x)
```

Arguments

`x`	a rectangular object

Value

a logical vector

Author(s)

Whit Armstrong

Examples

```
x <- fts(seq(from=Sys.Date(),by="months",length.out=50),matrix(rnorm(100),nrow=50))
jj <- x > 0
row.all(jj)
row.any(jj)

col.any(x > 0)
col.all(x > -3)
```

---

indicators *Trading indicators*

Description

Various binary indicators for fts objects

Usage

```
above.ma(x,n)
below.ma(x,n)
wday(x)
mday(x)
## S3 method for class 'fts'
diff(x,k,...)
up(x)
down(x)
ema(x,periods)
gap.continue(x)
gap.direction(x)
gap.down(x)
gap.down.continue(x)
gap.down.reverse(x)
gap.reverse(x)
gap.up(x)
gap.up.continue(x)
gap.up.reverse(x)
higher.high(x)
higher.low(x)
hl.oc.ratio(x)
inside.day(x)
inside.day.direction(x)
inside.day.down(x)
inside.day.up(x)
lower.high(x)
lower.low(x)
ma.crossover(x,n)
ma.crossover.down(x,n)
```

```
ma.crossover.up(x,n)
ma.d(x,n)
ma.distance(x,periods)
month(x)
year(x)
monthly.sum(x)
new.high(x,n)
new.low(x,n)
outside.day(x)
outside.day.direction(x)
outside.day.down(x)
outside.day.up(x)
pct.chg(x)
repeated(x,times)
rsi(x,periods)
rsi.crossover(x,periods,thresh)
rsi.crossover.down(x,periods,thresh)
rsi.crossover.up(x,periods,thresh)
template.fts(index, cnames)
trend.day(x,thresh)
trend.day.down(x,thresh)
trend.day.up(x,thresh)
```

Arguments

`x`	An Fts object
`periods`	number of periods
`n`	number of periods
`k`	number of lags
`times`	how many times
`thresh`	threshold level
`index`	index to use to construct fts object
`cnames`	colnames to use to construct fts object
`...`	further arguments to function

Details

removed elements are replaced with NA

Value

an fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="months",length.out=5),rnorm(5))
print(x)
lag(x,1)
lead(x,1)
```

---

intersect.all *find date intersection among multiple fts objects*

Description

find date intersection

Usage

```
intersect.all(...)
```

Arguments

...	Fts objects

Value

a vector of dates

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),data=1:100)
y <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),data=1:100)
y <- y[1:nrow(y) %% 2==0,]
intersect.all(x,y)
```

`lead.lag` *Shift an Fts ojbect in time*

Description

Shift an Fts ojbect forward or backwards in time be the supplied number of periods

Usage

```
## S3 method for class 'fts'
lead(x, k, ...)
## S3 method for class 'fts'
lag(x, k, ...)
```

Arguments

`x`	An Fts object
`k`	number of periods to shift
`...`	further arguments to function

Details

removed elements are replaced with NA

Value

an Fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=10),data=1:10)
print(x)
lag(x,1)
lead(x,1)
```

`moving` *Moving Functions*

Description

apply summary functions on a moving/rolling basis

Usage

```
moving.mean(x, periods)
moving.sum(x, periods)
moving.max(x, periods)
moving.min(x, periods)
moving.sd(x, periods)
moving.rank(x, periods)

moving.cor(x, y, periods)
moving.cov(x, y, periods)

cor.by.row(x,y)
```

Arguments

`x`	An Fts object
`y`	An Fts object
`periods`	integer: number of periods in window

Details

apply a function that takes a vector and returns a scalar on a rolling basis to an fts object.

For cor.by.row, the indicator is not rolling, but is the result of the application of the cor function to matching rows of x and y.

asking for a window larger than the number of rows of the fts object will result in an fts of all NA w/ the same number of rows as the input

for functions that take two fts objects the date intersection is taken before the window function is applied

Value

an fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),data=1:100)
y <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),data=1:100)

x.mean <- moving.mean(x,20)
x.sum <- moving.sum(x,20)
x.prod <- moving.product(x,20)
x.max <- moving.max(x,20)
x.min <- moving.min(x,20)
x.sd <- moving.sd(x,20)
x.rank <- moving.rank(x,20)

## take only odd rows
## to illustrate that teh correlation and covariance
## will only be calculated for the intersection of the dates
y <- y[(1:nrow(y))%%2 == 1]

xy.cor <- moving.cor(x, y, 20)
xy.cov <- moving.cov(x, y, 20)
```

---

pad *pad and trim dates*

Description

add dates to an Fts object by padding w/ additional dates or remove dates from an Fts object by trimming dates

Usage

```
pad(x, pad.dates)
trim(x, trim.dates)
filter.min.obs(x, obs.required)
```

Arguments

`x`	An Fts object
`pad.dates`	a vector of dates.
`trim.dates`	a vector of dates.
`obs.required`	number of required observations per row.

Value

an fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=5),data=1:5)
pad.dates <- index(x)[1] + c(10L,20L)
pad(x,pad.dates)

trim.dts <- index(x)[c(1,3)]

trim(x,trim.dts)
```

`read.write.fts` *Read / Write Files*

Description

Read / Write files to csv or .RDS format

Usage

```
read.csv.fts(file, date.column=1, date.format="%Y-%m-%d",
date.convert.fun=as.Date, ...)
write.csv.fts(x, file, ...)
```

Arguments

`x`	An Fts object
`file`	filename of file to read/write
`date.column`	column that = the dates are in
`date.format`	the format of the date strings
`date.convert.fun`	function to convert dates into desired index class
`...`	further arguments to underlying read/write functions

Value

a Fts object for functions that read data

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),data=1:100)
colnames(x) <- "big.ass.black.dog"

csv.fname <- paste(tempfile(),".csv",sep="")
write.csv.fts(x,csv.fname)
y.csv <- read.csv.fts(csv.fname)

all.equal(x,y.csv)
```

---

remove.rows *Remove Rows*

Description

remove.na.rows removes rows which contain at least 1 NA remove.all.na rows removes rows which are all NA's

Usage

```
remove.na.rows(x)
remove.all.na.rows(x)
```

Arguments

x	An Fts object

Value

an Fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=10),matrix(rnorm(20),ncol=2))

x[5,1] <- NA
x[10,] <- NA

print(x)

## will drop rows where NA's appear
## in any of the columns
remove.na.rows(x)

## will drop rows where NA's appear
```

```
## in all of the columns
remove.all.na.rows(x)
```

`since.na` *Count distance since an NA has occurred*

Description

Count number of rows since an NA has occurred

Usage

```
since.na(x)
```

Arguments

`x`	An Fts object

Value

an Fts object

Author(s)

Whit Armstrong

Examples

```
x <- fts(index=seq(from=Sys.Date(),by="days",length.out=100),rnorm(100))

x[10,] <- NA

since.na(x)
```

Index