Package ‘CNLTtsa’

July 18, 2018

Type Package

Title Complex-Valued Wavelet Lifting for Univariate and Bivariate Time Series Analysis

Version 0.1-2

Date 2018-07-18

Author Jean Hamilton [aut],
Matt Nunes [aut, cre],
Marina Knight [ctb],
Piotr Fryzlewicz [ctb]

Maintainer Matt Nunes <nunesrpackages@gmail.com>

Description Implementations of recent complex-valued wavelet spectral procedures for analysis of irregularly sampled signals, see Hamilton et al (2018) <doi:10.1080/00401706.2017.1281846>.

License GPL-2

Depends R(>= 3.1), adlift, nlt, CNLTreg, fields

NeedsCompilation no

Repository CRAN

Date/Publication 2018-07-18 13:00:11 UTC

R topics documented:

CNLTtsa-package	*Complex-Valued Wavelet Lifting for Univariate and Bivariate Time Series Analysis*

Description

Implementations of recent complex-valued wavelet spectral procedures for analysis of irregularly sampled signals, see Hamilton et al (2018) <doi:10.1080/00401706.2017.1281846>.

Details

The DESCRIPTION file:

Package:	CNLTtsa
Type:	Package
Title:	Complex-Valued Wavelet Lifting for Univariate and Bivariate Time Series Analysis
Version:	0.1-2
Date:	2018-07-18
Author:	Jean Hamilton [aut], Matt Nunes [aut, cre], Marina Knight [ctb], Piotr Fryzlewicz [ctb]
Authors@R:	c(person("Jean", "Hamilton", role="aut"),person("Matt","Nunes", role=c("aut","cre"),email="nunesrpackages
Maintainer:	Matt Nunes <nunesrpackages@gmail.com>
Description:	Implementations of recent complex-valued wavelet spectral procedures for analysis of irregularly sampled sign
License:	GPL-2
Depends:	R(>= 3.1), adlift, nlt, CNLTreg, fields

Index of help topics:

```
Baidu                   End of second returns for Google from 1st March
                        2011
CNLTtsa-package         Complex-Valued Wavelet Lifting for Univariate
                        and Bivariate Time Series Analysis
Google                  End of second returns for Google from 1st March
                        2011
cnlt.biv                Performs 'nondecimated' complex-valued wavelet
                        lifting for bivariate time series analysis
cnlt.spec               A function to compute CNLT spectral quantities
                        for univariate and bivariate series
cnlt.univ               Performs 'nondecimated' complex-valued wavelet
                        lifting for univariate time series analysis
cnltspec.plot           A function to plot CNLT spectral quantities of
                        interest
pre.per                 Functions to form periodogram objects with a
                        common time and scale bins for bivariate series
                        with different sampling grids for each
                        component
smooth.over.scale       Function to perform smoothing over scale of
```

	spectral quantities
smooth.over.time	Function to perform smoothing over time of spectral quantities

The main routines of the package are cnlt.univ and cnlt.biv which perform the nondecimated complex-valued lifting transform for univariate and bivariate time series, respectively; cnlt.spec computes spectral quantities of interest, for example the coherence and phase between two components of a bivariate series.

Author(s)

Jean Hamilton [aut], Matt Nunes [aut, cre], Marina Knight [ctb], Piotr Fryzlewicz [ctb]

Maintainer: Matt Nunes <nunesrpackages@gmail.com>

References

Hamilton, J., Nunes, M. A, Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

For related literature on the lifting methodology adopted in the technique, see

Knight, M. I. and Nason, G. P. (2009) A 'nondecimated' wavelet transform. *Stat. Comput.* **19** (1), 1–16.

Knight, M. I., Nunes, M. A. and Nason, G. P. (2012) Spectral Estimation for Locally Stationary Time Series with Missing Observations. *Stat. Comput.* **22** (4), 877–895.

See Also

cnlt.reg

Baidu *End of second returns for Google from 1st March 2011*

Description

Often several trades per second of a stock occur; this dataset consists of the last quoted value for each second for 1st March 2011. Thus the finest sampling interval is one second, but as there are seconds with no trades, the data have an unequally spaced sampling regime.

Usage

```
data("Baidu")
```

Format

A data frame with 7984 observations on the following 3 variables.

`Time` A variable with the time of the trade.

`Seconds.index` An index representing the time (in seconds) from the start of the data, representing the sampling regime of the series.

`Return` The return price of the stock.

References

Hamilton, J., Nunes, M. A, Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

Examples

```
data(Baidu)

plot(Baidu$Seconds.index,Baidu$Return,type="l")
```

---

cnlt.biv — *Performs 'nondecimated' complex-valued wavelet lifting for bivariate time series analysis*

Description

The forward complex-valued lifting transform for decomposing a signal of interest is dependent on the trajectory (lifting order) used in the forward lifting transform. This procedure uses trajectory bootstrapping to provide (complex-valued) time-scale information at all times and scales for bivariate series

Usage

```
cnlt.biv(x1, x2 = NULL, f1, f2, P = 100, nkeep = 2, use.same.trajectories = FALSE,
verbose = TRUE, ...)
```

Arguments

`x1`	A vector of grid values. Can be of any length, not necessarily equally spaced.
`x2`	An optional vector of grid values corresponding to `f2`. Can be of any length, not necessarily equally spaced. If not specified (`NULL`), then the same grid is used for `f2` as `f1`, i.e. `x1`.
`f1`	A vector of function values of the first component of a bivariate series, corresponding to `x`. Must be of the same length as `x`.
`f2`	A vector of function values of the second component of a bivariate series, corresponding to `x`. Must be of the same length as `x`.
`P`	Number of trajectories to be used in the nondecimated lifting transform.

nkeep	Number of scaling points we want at the end of the transform. The usual choice is nkeep=2.
use.same.trajectories	A boolean variable indicating whether the same set of trajectories should be used for both components of the bivariate signal.
verbose	Indicates whether useful messages should be printed to the console during the procedure.
...	Any other arguments to be passed to fwtnppermC, see the function documentation for more details.

Details

Essentially, this function applies the forward complex wavelet lifting transform fwtnppermC P times on both (x,f1) and (x,f2), each with a different random lifting trajectory. This results in P sets of complex-valued detail coefficients, along with their associated scales. This information is stored in order to compute the cross-periodograms for the bivariate series (x,f1,f2). The "degree of asymmetry" in the prediction is also recorded. This is the ratio between the maximum distance to the removed point to one of its neighbours and the minimum distance from the removed point to one of its neighbours, see Chapter 5.3 in Sanderson (2010) for more details.

Value

An object of class cnlt (subclasses biv and either SG or DG).
If both components have the same grid (subclass SG), a list with components:

x1	The sampling grid corresponding to f1 used for the decomposition.
x2	The sampling grid corresponding to f2 used for the decomposition. If the object is of subclass SG, x1 is the same as x2.
det1	A list, entry i corresponding to detail coefficients associated to point x_i and f1.
det2	A list, entry i corresponding to detail coefficients associated to point x_i and f2.
lre	A list, entry i corresponding to the scales (integrals) when lifting point x_i and f1.
lreA	A list, entry i corresponding to the degree of asymmetry of the neighbourhood used in the prediction step of point x_i, see description above.

If both components have different sampling grids, the additional following list components are returned:

lre2	A list, entry i corresponding to the scales (integrals) when lifting point x_i2 and f2.
lreA2	A list, entry i corresponding to the degree of asymmetry of the neighbourhood used in the prediction step of point x_i2 with f2, see description above.

Warning

Using a large number of trajectories for long datasets could take a long time!

Author(s)

Matt Nunes, Jean Hamilton

References

Hamilton, J., Nunes, M. A., Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

Sanderson, J. (2010) Wavelet methods for time series with bivariate observations and irregular sampling grids. PhD Thesis, University of Bristol, UK.

For the real-valued equivalent procedure, see also
Knight, M. I., Nunes, M. A. and Nason, G. P. (2012) Spectral Estimation for Locally Stationary Time Series with Missing Observations. *Stat. Comput.* **22** (4), 877–895.

See Also

`fwtnppermC`, `link{cnlt.univ}`

Examples

```
# a bivariate series example with same grids

# simulate data, e.g. two sinusoids
dat <- seq(from=1, to=3, by=0.1)
x1 <- cumsum(sample(dat, 200, TRUE))

y1 <-sin(2*pi*(1/25)*x1) + sin(2*pi*(1/50)*x1)+ 1*sin(2*pi*(1/10)*x1)+ rnorm(length(x1), 0,0.2)
y3 <- c(sin(2*pi*(1/25)*x1[1:100]),sin(2*pi*(1/25)*x1[97:196]))+ rnorm(length(x1), 0,0.1)

## Not run:
y1y3.dec<-cnlt.biv(x1, f1=y1, f2=y3, P = 500)

# the complex detail coefficients corresponding to the first timepoint are:

y1y3.dec$det1[[1]]


## End(Not run)

# a bivariate series example with different grids

# load some data in

data(Baidu)
data(Google)
```

```
## Not run:
BaiGoo<-cnlt.biv(Baidu$Seconds[1:100], Google$Seconds[1:100], Baidu$Return[1:100],
Google$Return[1:100], P = 500)

# now look at some of the coefficients from the decomposition
# (the complex detail coefficients corresponding to the first timepoint:

BaiGoo$det1[[1]]
BaiGoo$det2[[1]]

## End(Not run)
```

`cnlt.spec` *A function to compute CNLT spectral quantities for univariate and bivariate series*

Description

The function takes a nondecimated complex lifting decomposition of a univariate or bivariate series, and uses smoothing before computing spectral quantities such as the complex periodograms, coherence and phase

Usage

```
cnlt.spec(x, ...)
## S3 method for class 'SG'
cnlt.spec(x, M = 50, fact = 1, ...)
## S3 method for class 'DG'
cnlt.spec(x, M = 50, fact = 1, ...)
```

Arguments

x	An object of class `cnlt`, i.e. the output from either `cnlt.univ` or `cnlt.biv`.
M	The smoothing parameter (binwidth) or vector of smoothing parameters (one for each scale) for the time-domain kernel smoothing method, see `smooth.over.time`.
fact	If `length(M)==1`, a factor indicating how the smoothing parameter (binwidth) in the time-domain kernel smoothing method should increase from one scale to the next, see `smooth.over.time`.
...	Any other parameters to be passed to the scale smoothing function, see the documentation for `smooth.over.scale` for univariate `cnlt` objects, or `pre.per` for bivariate `cnlt` objects.

Details

For univariate series, the nondecimated complex lifting object can be used to form a spectral object by smoothing the squared details over scale (with `smooth.over.scale`), and then smoothing over time (using `smooth.over.time`). Smoothing over scale is done via `smooth.spline`; smoothing over time is done with a kernel smoother (e.g. a "box" kernel for a moving average). See Hamilton et al. (2018) for more details.

Value

An object of class `cnlt.spec` (subclasses: `DG`, `SG`, `univ`, `biv`).

For subclass `univ`, a list with components:

`S1`	A spectral object (matrix) of dimension `length(mscale) x length(mtime)`, corresponding to the spectrum of the univariate series.
`mscale`	A vector of scales corresponding to the rows of the spectrum `S1` (after smoothing the periodogram), see `smooth.over.scale`.
`mtime`	The vector `cnltobj$x`, the vector of times corresponding to the columns of the spectrum `S1`.

For subclass `biv`, a list with components:

`coh`	A matrix of dimension `length(mscale) x length(mtime)`, corresponding to the coherence between the two components of the bivariate series.
`phase`	A matrix of dimension `length(mscale) x length(mtime)`, corresponding to the phase between the two components of the bivariate series.
`C`	A matrix of dimension `length(mscale) x length(mtime)`, corresponding to the co-periodogram of the bivariate series.
`Q`	A matrix of dimension `length(mscale) x length(mtime)`, corresponding to the quadrature periodogram of the bivariate series.
`S1`	A matrix of dimension `length(mscale) x length(mtime)`, corresponding to the spectrum of the first component of the bivariate series.
`S2`	A matrix of dimension `length(mscale) x length(mtime)`, corresponding to the spectrum of the second component of the bivariate series.
`mscale`	A vector of scales corresponding to the rows of the spectrum `S1` (after smoothing the periodogram), see `smooth.over.scale`.
`mtime`	A vector of times corresponding to the columns of the spectrum `S1`. If the class of `cnlt.obj` is `SG`, this is `cnlt.obj$x1`, else this is a vector formed by binning detail coefficients within equal intervals of time, see `pre.per` for more details.

Author(s)

Matt Nunes, Jean Hamilton

References

Hamilton, J., Nunes, M. A., Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

See Also

cnlt.biv, cnlt.univ, cnltspec.plot

Examples

```
# read some data in (a bivariate series)

## Not run:

data(Baidu)
data(Google)

BaiGoo<-cnlt.biv(Baidu$Seconds[1:100], Google$Seconds[1:100], Baidu$Return[1:100],
Google$Return[1:100], P = 500)

specobj<-cnlt.spec(BaiGoo,M=10,fact=1.05, Tstar=20)

## End(Not run)
```

---

cnlt.univ — *Performs 'nondecimated' complex-valued wavelet lifting for univariate time series analysis*

Description

The forward complex-valued lifting transform for decomposing a signal of interest is dependent on the trajectory (lifting order) used in the forward lifting transform. This procedure uses trajectory bootstrapping to provide (complex-valued) time-scale information at all times and scales

Usage

```
cnlt.univ(x, f, P = 100, nkeep = 2, verbose = TRUE, ...)
```

Arguments

x	A vector of grid values. Can be of any length, not necessarily equally spaced.
f	A vector of function values corresponding to x. Must be of the same length as x.
P	Number of trajectories to be used in the nondecimated lifting transform.
nkeep	Number of scaling points we want at the end of the transform. The usual choice is nkeep=2.

verbose	Indicates whether useful messages should be printed to the console during the procedure.
...	Any other arguments to be passed to fwtnppermC, see the function documentation for more details.

Details

Essentially, this function applies the forward complex wavelet lifting transform fwtnppermC P times, each with a different random lifting trajectory. This results in P sets of complex-valued detail coefficients, along with their associated scales. This information is stored in order to compute the periodogram for (x,f). The “degree of asymmetry" in the prediction is also recorded. This is the ratio between the maximum distance to the removed point to one of its neighbours and the minimum distance from the removed point to one of its neighbours, see Chapter 5.3 in Sanderson (2010) for more details.

Value

An object of class cnlt (subclasses: univ, SG). A list with components:

x	The sampling grid corresponding to f used for the decomposition.
det1	A list, entry i corresponding to detail coefficients associated to point x_i.
lre	A list, entry i corresponding to the scales (integrals) when lifting point x_i.
lreA	A list, entry i corresponding to the degree of asymmetry of the neighbourhood used in the prediction step of point x_i, see description above.

Warning

Using a large number of trajectories for long datasets could take a long time!

Author(s)

Jean Hamilton, Matt Nunes

References

Hamilton, J., Nunes, M. A., Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

Sanderson, J. (2010) Wavelet methods for time series with bivariate observations and irregular sampling grids. PhD Thesis, University of Bristol, UK.

For the real-valued equivalent procedure, see also
Knight, M. I., Nunes, M. A. and Nason, G. P. (2012) Spectral Estimation for Locally Stationary Time Series with Missing Observations. *Stat. Comput.* **22** (4), 877–895.

See Also

fwtnppermC, link{get.percoeffsC.biv}

Examples

```
x<-sort(runif(100))

y <-sin(2*pi*(1/25)*x) + sin(2*pi*(1/50)*x)

## Not run:
xy.dec<-cnlt.univ(x,y,P=300)

xy.dec$det[[1]][1:10]

## End(Not run)
```

`cnltspec.plot` *A function to plot CNLT spectral quantities of interest*

Description

The function takes a spectral quantity and plots it according to user inputted graphical options

Usage

```
cnltspec.plot(spec, timevec, scalevec, zrange = NULL, xtitle = "Time", ytitle = "Scale",
col.scale = tim.colors(64)[1:45], SFratio = 2, dt = 1, parsw = 3, axis4 = FALSE,
frequencies = NULL)
```

Arguments

`spec`	A spectral quantity, i.e. contained in a `cnlt.spec` object. For example, this could be the coherence between components of a bivariate series.
`timevec`	A vector corresponding to the x-axis of the spectral object, often time or a sampling grid.
`scalevec`	A vector of scales corresponding to the y-axis of the spectral object.
`zrange`	An optional length two vector specifying the range of the z-axis of the plot.
`xtitle`	A label for the x-axis of the plot.
`ytitle`	A label for the y-axis of the plot.
`col.scale`	a color palette to use for the spectral plot.
`SFratio`	A number specifying the relationship between scale and Fourier frequency, see `frequencies` argument, and Sanderson (2010), chapter 6.2.
`dt`	A sampling interval, used to compute the relationship between scale and Fourier frequency, see Sanderson (2010), chapter 6.2.
`parsw`	A number from 1 to 3, specifying different spacings between the plot and the legend. This is useful if you want to do call `cnltspec.plot` multiple times for e.g. multi-panel plots.

`axis4` An optional boolean variable indicating whether a 4th (right) axis should be added to the plot.

`frequencies` If `axis4 = TRUE`, an optional vector for the ticks on the 4th axis. If these are not specified, then a vector of Fourier frequencies are plotted, with the relationship between scale and frequency specified by `SFratio`, see Sanderson (2010), chapter 6.2.

Value

A spectral quantity is plotted.

Author(s)

Jean Hamilton, Matt Nunes

References

Hamilton, J., Nunes, M. A., Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

Sanderson, J. (2010) Wavelet methods for time series with bivariate observations and irregular sampling grids. PhD Thesis, University of Bristol, UK.

See Also

`cnlt.spec`

Examples

```
# simulate data, e.g. two sinusoids
dat <- seq(from=1, to=3, by=0.1)
x1 <- cumsum(sample(dat, 200, TRUE))

y1 <-sin(2*pi*(1/25)*x1) + sin(2*pi*(1/50)*x1)+ 1*sin(2*pi*(1/10)*x1)+ rnorm(length(x1), 0,0.2)
y3 <- c(sin(2*pi*(1/25)*x1[1:100]),sin(2*pi*(1/25)*x1[97:196]))+ rnorm(length(x1), 0,0.1)

## Not run:

y1y3.dec<-cnlt.biv(x1, f1=y1, f2=y3, P = 500)

y1y3spec<-cnlt.spec(y1y3.dec)

cnltspec.plot(y1y3spec$coh,y1y3spec$mtime,y1y3spec$mscale)

## End(Not run)
```

Google *End of second returns for Google from 1st March 2011*

Description

Often several trades per second of a stock occur; this dataset consists of the last quoted value for each second for 1st March 2011. Thus the finest sampling interval is one second, but as there are seconds with no trades, the data have an unequally spaced sampling regime.

Usage

```
data("Google")
```

Format

A data frame with 6526 observations on the following 3 variables.

`Time` A variable with the time of the trade.

`Seconds.index` An index representing the time (in seconds) from the start of the data, representing the sampling regime of the series.

`Return` The return price of the stock.

References

Hamilton, J., Nunes, M. A, Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

Examples

```
data(Google)

plot(Google$Seconds.index,Google$Return,type="l")
```

pre.per *Functions to form periodogram objects with a common time and scale bins for bivariate series with different sampling grids for each component*

Description

The CNLT forms detail coefficients for each component of a bivariate series. Due to the two components having different sampling grids, the details (and associated scales) won't have a common association for both series. Hence the details are sampled and mapped to a common timescale and a common set of scales via binning and averaging. These functions compute spectral objects on these common times / scales

Usage

```
pre.per(x, det, lre, lreA, scale.range = NULL, time.range = NULL, Arange = NULL,
Jstar = 20, Tstar = 50)
pre.per.comb(spec1, spec2)
pre.per.sample(spec1, spec2)
```

Arguments

x	A vector corrsponding to the sampling grid of a component of a series.
det	A list of (real or imaginary parts of) the detail coefficients from a CNLT decomposition, such as from the output of cnlt.biv.
lre	A list of scales (removed integral lengths) corresponding to det from a CNLT decomposition, such as from the output of cnlt.biv.
lreA	A list of asymmetry values from a CNLT decomposition, such as from the output of cnlt.biv.
scale.range	An optional two-vector specifying the range of scales to be considered in the resulting output spectrum.
time.range	An optional two-vector specifying the range of times to be considered in the resulting output spectrum.
Arange	An optional two-vector specifying whether the values used in forming the output spectrum should be limited to those from a specific range of asymmetry values, see Sanderson (2010), chapter 6.2.
Jstar	The number of artificial scales in the output spectrum.
Tstar	The number of artificial times in the output spectrum.
spec1	A periodogram corresponding to the first component of a bivariate series.
spec2	A periodogram corresponding to the second component of a bivariate series.

Details

For a bivariate series where the two components have different sampling grids, the co- /quadrature periodogram values are first formed using `pre.per`, using a vector of `Tstar` equal time intervals, specified by setting `Tstar` and optionally `time.range`; they are also binned into `Jstar` equal artificial levels by setting `Jstar` and optionally `scale.range`. The details are sampled using a common sampling vector with `pre.per.sample`, and then combined using `pre.per.comb`. The periodogram is then smoothed over time. See Hamilton et al (2018), section 2.3 for more details.

Value

For `pre.per`, a list with components:

spec	A matrix of dimension `Jstar x Tstar` corrsponding to a quadrature periodogram / co-periodogram of a bivariate series.
mscale	A vector of scales (of length `Jstar`) corresponding to the rows of the spectrum `spec`.

mtime — A vector of times (of length Tstar) corresponding to the columns of the spectrum spec.

For pre.per.sample, a list with components:

spec1 — A matrix of dimension Jstar x Tstar corrsponding to a periodogram of the first component of a bivariate series.

spec2 — A matrix of dimension Jstar x Tstar corrsponding to a periodogram of the second component of a bivariate series.

For pre.per.comb, a list with components:

spec — A matrix of dimension Jstar x Tstar corrsponding to a periodogram / quadrature periodogram / co-periodogram of a bivariate series.

Note

Note that these functions aren't intended to be used directly, but are called internally from the function cnlt.spec.DG.
Note also that the argument Tstar should be chosen small enough so that the range of the sampling grid x can be divided into equally spaced intervals, with *at least one* gridpoint in an interval.

Author(s)

Jean Hamilton, Matt Nunes

References

Hamilton, J., Nunes, M. A., Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

See Also

cnlt.spec.DG, smooth.over.time

Examples

```
# simulate data, e.g. two sinusoids
dat <- seq(from=1, to=3, by=0.1)
x1 <- cumsum(sample(dat, 200, TRUE))
x2 <- cumsum(sample(dat, 200, TRUE))

y1 <-sin(2*pi*(1/25)*x1) + sin(2*pi*(1/50)*x1)+ 1*sin(2*pi*(1/10)*x1)+ rnorm(length(x1), 0,0.2)
y3 <- sin(2*pi*(1/25)*x2[97:196]) + rnorm(length(x2), 0,0.1)

## Not run:
y1y3.dec<-cnlt.biv(x1, f1=y1, f2=y3, P = 500)

# compute the co-periodogram for the first component...
```

```
C1 <- pre.per(x1, sapply(y1y3.dec$det1,Re), y1y3.dec$lre1, y1y3.dec$lreA1, Jstar = 10)

# .. and for the second component
C2 <- pre.per(x1, sapply(y1y3.dec$det2,Re), y1y3.dec$lre2, y1y3.dec$lreA2, Jstar = 10)

## End(Not run)
```

`smooth.over.scale` *Function to perform smoothing over scale of spectral quantities*

Description

This function uses simple averaging or smoothing splines to smooth spectra over scale

Usage

```
smooth.over.scale(x, det1, det2, lre, lreA, scale.range = NULL, Arange = NULL,
Jstar = 20, splines = FALSE, positive = FALSE, dfS = 10, interpolate = FALSE)
```

Arguments

`x`	A vector corrsponding to the sampling grid of the component of a univariate series, or both components of a bivariate series with identical sampling grids.
`det1`	A list of (real or imaginary parts of) the component 1 detail coefficients from a CNLT decomposition, such as from the output of `cnlt.biv`.
`det2`	A list of (real or imaginary parts of) the component 2 detail coefficients from a CNLT decomposition, such as from the output of `cnlt.biv`.
`lre`	A list of scales (removed integral lengths) corresponding to `det` from a CNLT decomposition, such as from the output of `cnlt.biv`.
`lreA`	A list of asymmetry values from a CNLT decomposition, such as from the output of `cnlt.biv`.
`scale.range`	An optional two-vector specifying the range of scales to be considered in the resulting output spectrum.
`Arange`	An optional two-vector specifying whether the values used in forming the output spectrum should be limited to those from a specific range of asymmetry values, see Sanderson (2010), chapter 6.2.
`Jstar`	The number of artificial scales in the output spectrum.
`splines`	An indicator variable whether smoothing splines should be used for the scale-based smoothing, or simple averaging (`splines = FALSE`).
`positive`	An indicator variable whether the smoothing should ensure that the resulting output is positive or not (e.g. for spectra).
`dfS`	An argument, if `splines = TRUE`, specifying the number of degrees of freedom for the smoothing spline.
`interpolate`	An indicator variable for whether interpolation should be used in the smoothing spline method for predicting values outside the range of the data.

Details

For a univariate series or a bivariate series where the two components have the same sampling grids, the co- /quadrature periodogram values are first formed. They are then smoothed over scale (per timepoint), to give spectral values corresponding to equal artificial levels by setting `Jstar` and optionally `scale.range`.

Value

A list with the following components:

`spec`	A matrix of dimension `Jstar x length(x)` corrsponding to a periodogram / co-periodogram / quadrature periodogram.
`mscale`	A vector of scales (of length `Jstar`) corresponding to the rows of the spectrum `spec`.

Author(s)

Jean Hamilton, Matt Nunes

References

Hamilton, J., Nunes, M. A., Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

See Also

`cnlt.spec.SG`

Examples

```
x<-sort(runif(100))

y <-sin(2*pi*(1/25)*x) + sin(2*pi*(1/50)*x)

## Not run:
xy.dec<-cnlt.univ(x,y,P=300)

# compute the real part of the spectrum (real details^2) and smooth over scale
ReS <- smooth.over.scale(x, sapply(xy.dec$det1,Re), sapply(xy.dec$det1,Re), xy.dec$lre,
xy.dec$lreA, positive = TRUE)

## End(Not run)
```

`smooth.over.time` *Function to perform smoothing over time of spectral quantities*

Description

This function uses a running mean (box kernel) smoother to smooth spectra over time, with potentially different smoothing parameters used for each scale of the spectra

Usage

```
smooth.over.time(x, spec, M, fact = 1)
```

Arguments

`x`	A vector corrsponding to the sampling grid of a component of a series.
`spec`	A spectral object (matrix), with rows corrsponding to different scales.
`M`	The smoothing parameter (binwidth) or vector of smoothing parameters (one for each scale) for the smoothing method.
`fact`	If `length(M)==1`, a factor indicating how the smoothing parameter (binwidth) in the time-domain kernel smoothing method should increase from one scale to the next.

Details

The function takes in a matrix and performs a kernel smoother on row `i` of the matrix, using a bandwidth of `M[i]` if `length(M)==nrow(spec)`, and `M * fact^{i-1}` if `length(M)==1`. Thus if the scaling factor, `fact`, is chosen to be greater than one, a wider kernel is used for the smoothing for later scales.

Value

`smooth.spec`	A matrix of same dimension as `spec`, containing smoothed spectral values.

Author(s)

Jean Hamilton

References

Hamilton, J., Nunes, M. A., Knight, M. I. and Fryzlewicz, P. (2018) Complex-valued wavelet lifting and applications. *Technometrics*, **60** (1), 48-60, DOI 10.1080/00401706.2017.1281846.

See Also

`cnlt.spec`, `pre.per`

Examples

```
x<-sort(runif(100))

y <-sin(2*pi*(1/25)*x) + sin(2*pi*(1/50)*x)

## Not run:
xy.dec<-cnlt.univ(x,y,P=300)

# compute the real part of the spectrum (real details^2) and smooth over scale, then over time
ReS <- smooth.over.scale(x, sapply(xy.dec$det1,Re), sapply(xy.dec$det1,Re), xy.dec$lre,
xy.dec$lreA, positive = TRUE)

ReS.smooth <- smooth.over.time(x,ReS$spec,5,1.05)

## End(Not run)
```

Index