Package ‘TrendLSW’

August 5, 2020

Type Package

Title Wavelet methods for analysing nonstationary time series

Version 0.1.1

Author Euan McGonigle

Maintainer Euan McGonigle <e.mcgonigle2@lancaster.ac.uk>

Description Wavelet-based routines for trend estimation and second-order estimation of nonstationary time series.

Encoding UTF-8

LazyData true

Imports wavethresh, locits, binhf

RoxygenNote 7.0.2

R topics documented:

Atau.mat.calc *Lagged Autocorrelation Wavelet Inner Product Calculation*

Description

Function that computes the matrix of lagged autocorrelation wavelet inner products.

Usage

```
Atau.mat.calc(J, filter.number = 1, family = "DaubExPhase", lag = 1)
```

Arguments

`J`	The dimension of the matrix required. Should be a positive integer.
`filter.number`	The index of the wavelet used to compute the inner product matrix.
`family`	The family of wavelet used to compute the inner product matrix.
`lag`	The lag of matrix to calculate. A lage of 0 corresponds to the standrad A matrix.

Details

This function computes the lagged inner product matrix of the discrete non-decimated autocorrelation wavelets. This matrix is used in the calculation to correct the wavelet periodogram of the differenced time series. The matrix returned is the one called A^tau in McGonigle et al. (2020).

Value

A J-dimensional square matrix giving the lagged inner product autocorrelation wavelet matrix.

Author(s)

E T McGonigle

References

McGonigle, E. T., Killick, R., and Nunes, M. (2020). Modelling Time-Varying First and Second-Order Structure of Time Series via Wavelets and Differencing.

Examples

```
Atau.mat.calc(J=5, filter.number = 1, family = "DaubExPhase", lag = 1)
```

calc.final.spec — *Auxilliary Function to Convert wd Objects*

Description

Auxiliary function used as a result of boundary handling. Not for general use.

Usage

```
calc.final.spec(spec)
```

Arguments

`spec`	The spectrum object based on the boundary corrected time series of length 4*length(data).

Details

Converts the boundary converted spectrum object back to its correct size.

Value

Spectrum object of class wd.

Author(s)

E T McGonigle

Cmat.calc — *Cross Autocorrelation Wavelet Inner Product Matrix Calculation*

Description

Function that computes the cross autocorrelation matrix of inner products.

Usage

```
Cmat.calc(J, gen.filter.number = 1, an.filter.number = 1,
                      gen.family = "DaubExPhase", an.family = "DaubExPhase")
```

Arguments

`J`	The dimension of the matrix required. Should be a positive integer.
`gen.filter.number`	The index of the generating wavelet used to compute the inner product matrix.
`an.filter.number`	The index of the analysing wavelet used to compute the inner product matrix.
`gen.family`	The family of generating wavelet used to compute the inner product matrix.
`an.family`	The family of analysing wavelet used to compute the inner product matrix.

Details

This function computes the cross inner product matrix of the discrete non-decimated autocorrelation wavelets. This matrix is used to correct the wavelet periodogram analysed using a diffferent wavelet to the wavelet that is assumed to generate the time series. The matrix returned is the one called C in McGonigle et al. (2020).

Value

A J-dimensional square matrix giving the cross inner product autocorrelation wavelet matrix.

Author(s)

E T McGonigle

References

McGonigle, E. T., Killick, R., and Nunes, M. (2020). Trend locally stationary wavelet processes with applications to environmental data. Submitted.

See Also

ewspec.trend, wav.diff.trend.est

Examples

```
Cmat.calc(J=5, gen.filter.number = 1, an.filter.number = 2,
gen.family = "DaubExPhase", an.family = "DaubExPhase")
```

---

create.covmat — *Create Covariance Matrix from lacf Object Function used to create the autocovariance matrix from an lacf object input.*

Description

This function used to create the autocovariance matrix from an lacf object input, which is used to calculate the confidence interval for the trend estimate.

Usage

```
create.covmat(lacf)
```

Arguments

lacf	An lacf object, which can be created from a spectral estimate using the function lacf.calc.

Value

A T x T autocovariance matrix.

Author(s)

E T McGonigle

See Also

lacf.calc, lacf

`easy.spec.plot` *Quick Plotting of Evolutionary Wavelet Spectrum*

Description

Plots a standard version of the EWS plot.

Usage

```
easy.spec.plot(spec, bylev = FALSE)
```

Arguments

`spec`	A spectrum object of class wd which you wish to plot
`bylev`	If TRUE, plots each level of the spectrum on its own individual scaling.

Author(s)

E T McGonigle

Examples

```
spec = cns(1024)
spec = putD(spec, level = 8, 1+sin(seq(from = 0, to = 2*pi,length = 1024))^2)

easy.spec.plot(spec)
```

`ewspec.diff` *Estimation of Evolutionary Wavelet Spectrum for Non-Zero Mean Time Series via Differencing.*

Description

This function computes the evolutionary wavelet spectrum (EWS) estimate from a time series that may include a trend component. The estimate is computed by taking the non-decimated wavelet transform of the first differenced time series data, squaring it; smoothing using a running mean and then correcting for bias using the appropriate correction matrix. Inherits the smoothing functionality from the ewspec3 function in the R package locits.

Usage

```
ewspec.diff(data, filter.number = 1, family = "DaubExPhase",
binwidth = floor(2 * sqrt(length(data))), diff.number = 1,
max.scale = floor(log2(length(data)) * 0.7), WP.smooth = TRUE,
AutoReflect = FALSE, supply.inv.mat = FALSE, inv = NULL)
```

Arguments

`data`	The time series you wish to analyse.
`filter.number`	The index number for the wavelet used to analyse the time series. For the "DaubExPhase" family, the filter number can be bewtween 1 to 10. For the "DaubLeAsymm" family, the filter number can be between 4 to 10.
`family`	The family of the wavelet used. It is recommended to use either the Daubechies Extremal Phase family, or the Daubechies Least Asymmetric family, corresponding to the "DaubExPhase" or the "DaubLeAsymm" options.
`binwidth`	The bin width of the running mean smoother used to smooth the raw wavelet periodogram.
`diff.number`	The number of differences used to remove the trend of the series. A first difference is reccomended as default.
`max.scale`	The coarsest level to which the time series is analysed to. Should be a positive integer less than J, where T=2^J is the length of the time series. The default setting is 0.7J, to control for bias.
`WP.smooth`	Argument that dictates if smoothing is performed on the raw wavelet periodogram.
`AutoReflect`	As in wavethresh. Decides whether or not the time series is reflected when computing the wavelet transform. Strongly recommended to leave as FALSE.
`supply.inv.mat`	Not intended for use. If TRUE, user must supply the appropriate correction matrix
`inv`	Not intended for use. If supply.mat is TRUE, user must supply the appropriate correction matrix used to correct the raw wavelet periodogram. Equal to solve(2*A-2*A1) for first differences.

Details

This function computes an estimate of the evolutionary wavelet spectrum of a time series that displays nonstationary mean and autocovariance. The estimation procedure is as follows:

1. The time series is first differenced to remove the trend.

2. The squared modulus of the non-decimated wavelet transform is computed, known as the raw wavelet periodogram. This is returned by the function.

3. The raw wavelet periodogram is smoothed using a running mean smoother.

4. The smoothed periodogram is bias corrected using the inverse of the bias matrix.

The final estimate, stored in the S component, can be plotted using the plot function, please see the example below.

Value

A list object, containing the following objects:

`S`	The evolutionary wavelet spectral estimate of the input data. This object is of class wd and so can be plotted and printed in the usual way using wavethresh functionality.
`WavPer`	The raw wavelet periodogram of the input data. The EWS estimate (above) is the smoothed corrected version of the wavelet periodgram.
`SmoothWavPer`	The smoothed, un-corrected raw wavelet periodogram of the input data.

Author(s)

E T McGonigle

References

McGonigle, E. T., Killick, R., and Nunes, M. (2020). Modelling Time-Varying First and Second-Order Structure of Time Series via Wavelets and Differencing.

See Also

ewspec, ewspec3, ewspec.trend

Examples

```
spec = cns(1024)
spec = putD(spec, level = 8, 1+sin(seq(from = 0, to = 2*pi,length = 1024))^2)

set.seed(2352)

noise = LSWsim(spec)
trend = c(seq(from = 0, to = 4,length = 400),seq(from = 4, to = 0,length = 1024))

x = trend+noise

spec.est = ewspec.diff(x,family = "DaubExPhase", filter.number = 1,max.scale = 7)

easy.spec.plot(spec.est$S)
```

---

ewspec.trend — *Estimation of Evolutionary Wavelet Spectrum for Non-Zero Mean Time Series*

Description

This function computes the evolutionary wavelet spectrum (EWS) estimate from a time series that may include a trend component. The estimate is computed by taking the non-decimated wavelet transform of the time series data, squaring it; smoothing using a running mean and then correction for bias using the appropriate correction matrix. Inherits the smoohing functionality from the ewspec3 function in thr R package wavethresh.

Usage

```
ewspec.trend(data, an.filter.number  = 10, an.family = "DaubLeAsymm",
gen.filter.number  = 10, gen.family = "DaubLeAsymm",
binwidth = floor(2*sqrt(length(data))),
max.scale = floor(log2(length(data))*0.7), WP.smooth = TRUE,
AutoReflect = TRUE, supply_mat = FALSE, mat = NULL,
boundary.handle = TRUE)
```

Arguments

- `data` The time series you wish to analyse.
- `an.filter.number` The index number for the wavelet used to analyse the time series. For the "DaubExPhase" family, the filter number can be bewtween 1 to 10. For the "DaubLeAsymm" family, the filter number can be between 4 to 10. Similarly for gen.filter.number.
- `an.family` The family of the analysing wavelet. It is recommended to use either the Daubechies Extremal Phase family, or the Daubechies Least Asymmetric family, corresponding to the "DaubExPhase" or the "DaubLeAsymm" options. Similarly for gen.family.
- `gen.filter.number` The index number for the wavelet that generates the stochastic component of the time series.
- `gen.family` The family of the generating wavelet.
- `binwidth` The bin width of the running mean smoother used to smooth the raw wavelet periodogram.
- `max.scale` The coarsest level to which the time series is analysed to. Should be a positive integer less than J, where T=2^J is the length of the time series. The default setting is 0.7J, to control for bias from the trend and boundary effects.
- `WP.smooth` Argument that dictates if smoothing is performed on the raw wavelet periodogram.
- `AutoReflect` As in wavethresh. Decides whether or not the time series is reflected when computing the wavelet transform. Helps estimation at the boundaries.
- `supply.mat` Not intended for use. If TRUE, user must supply the appropriate correction matrix
- `mat` If supply.mat is TRUE, user must supply the appropriate correction matrix used to correct the raw wavelet periodogram. Equal to solve(C).
- `boundary.handle` Can be TRUE or FALSE. If TRUE, the time series is boundary corrected, to get a more accurate spectrum estimate at the boundaries of the times series. If FALSE, no boundary correction is applied. Recommned to use TRUE.

Details

This function computes an estimate of the evolutionary wavelet spectrum of a time series that displays a smooth mean and nonstationary autocovariance. The estimation procedure is as follows:

1. The squared modulus of the non-decimated wavelet transform is computed, known as the raw wavelet periodogram. This is returned by the function.

2. The raw wavelet periodogram is smoothed using a running mean smoother.

3. The smoothed periodogram is bias corrected using the inverse of the bias matrix. The correction is applied across the finest max.scale scales. If the analyising wavelet and generating wavelet are different, this is given by the iverse of the C matrix defined in McGonigle et al. (2020). If they are the same, this is the inverse of the A matrix, defined in Nason et al. (2000). If you are unsure on the filter and wavelet choices, it is recommended to use the same wavelet for generating and analysing purposes.

The final estimate, stored in the S component, can be plotted using the plot function, please see the example below.

Value

A list object, containing the following objects:

S	The evolutionary wavelet spectral estimate of the input data. This object is of class wd and so can be plotted and printed in the usual way using wavethresh functionality.
WavPer	The raw wavelet periodogram of the input data. The EWS estimate (above) is the smoothed corrected version of the raw wavelet periodgram.
SmoothWavPer	The smoothed, un-corrected raw wavelet periodogram of the input data.

Author(s)

E T McGonigle

References

McGonigle, E. T., Killick, R., and Nunes, M. (2020). Trend locally stationary wavelet processes with applications to environmental data. Submitted.

Nason, G. P., von Sachs, R., and Kroisandt, G. (2000). Wavelet processes and adaptive estima- tion of the evolutionary wavelet spectrum. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 62(2):271–292.

Examples

```
##---- simulates an example time series and estimates its EWS


spec = cns(1024)
spec = putD(spec, level = 8, 1+sin(seq(from = 0, to = 2*pi,length = 1024))^2)

noise = LSWsim(spec)
trend = seq(from = 0, to = 10,length = 1024)

x = trend+noise

spec.est = ewspec.trend(x, an.filter.number =4, an.family = "DaubExPhase", gen.filter.number = 1, gen.family

easy.spec.plot(spec.est$S)
```

---

get.boundary.timeseries

Calculates Boundary Extended Time Series

Description

A function to calclate the boundary extended time series, to be used within the ewspec.trend and wav.trend.est functions.

Usage

```
get.boundary.timeseries(data)
```

Arguments

`data`	The time series used to calculate the boundary extended version.

Value

A vector of 4 times the length of the input vector.

Author(s)

E T McGonigle

`lacf.calc` *Create lacf Object from Spectrum Estimate*

Description

Function that produces an lacf object that contains the local autocovariance and autocorrelation estimates, given an input of a spectrum estimate. Same functionality as lacf function from locits, but user specifies the spectrum estimate in the argument.

Usage

```
lacf.calc(x)
```

Arguments

	Only additional argument is
	Spectrum from which the lacf estimate will be calculated.

Author(s)

`altered_spec` E T McGonigle

See Also

`lacf`

`LSWsim.anydist` *Simulate locally stationary wavelet processes with random innovations not necessarily Gaussian.*

Description

Simulates a locally stationary wavelet process given a spectrum and distribution for the random innovations. Extension of the LSWsim function from wavethresh.

Usage

```
LSWsim.anydist(spec, distribution = "Normal")
```

Arguments

`spec`	An object of class wd which contains the spectrum for simulating an LSW process.
`distribution`	The distribution of the random variables used to simulate the process. Can be "Normal", "Exponential", "Chisquare" or "Poisson".

Value

A vector simulated from the spectral description given in the spec description. The returned vector will exhibit the spectral characteristics defined by spec.

Author(s)

E T McGonigle

See Also

`LSWsim`

Examples

```
spec = cns(1024)

spec = putD(spec,level = 8, seq(from = 2, to = 8,length = 1024))

x = LSWsim.anydist(spec, distribution = "Exponential")

plot.ts(x)
```

`trend.estCI` *Calculate Confidene Intervals Of Wavelet-Based Trend Estimate*

Description

A function to calculate appropriate confidence intervals for the trend estimate, based on a given trend estimate and covariance matrix of the time series.

Usage

```
trend.estCI(trend_est, cov_mat, filter.number, family, alpha)
```

Arguments

`trend_est`	The trend estimate of the time series.
`cov_mat`	The covariance matrix estimate of the time series.
`filter.number`	The index of the wavelet used in the estimation procedure.
`family`	The wavelet family used in the estimation procedure.
`alpha`	The significance level of the confidence interval to calculate. Default is 0.95.

Value

A list object, containing the following objects:

`trend.est`	The trend estimate of the input data.
`trend.var`	The variance estimate of the trend estimate.
`upper.conf`	The upper pointwise confidence interval for the trend estimate.
`lower.conf`	The lower pointwise confidence interval for the trend estimate.

Author(s)

E T McGonigle

See Also

`wav.trend.est`, `lacf.calc`, `create.covmat` `wav.diff.trend.est` *Wavelet Thresholding Trend Estimation of Second-Order Nonstationary Time Series*

Description

This function computes the wavelet thresholding trend estimate for a time series that may be second-order nonstationary. The function calculates the wavelet transform of the time series, thresholds the coefficients based on an estimate of their variance, and inverts to give the trend estimate.

Usage

```
wav.diff.trend.est(data, spec, filter.number = 4, thresh.type = "soft",
normal = TRUE, family = "DaubLeAsymm", max.scale = floor(0.7 * log2(length(data))),
trans.type = "nondec")
```

Arguments

`data`	The time series you want to estimate the trend function of.
`spec`	You must supply the estimate of the evolutionary wavelet spectrum of the time series. This is calculated using the function ewspec.diff, and selecting the S component from the returned list object.
`filter.number`	Selects the index of the wavelet used in the estimation procedure. For Daubechies compactly supported wavelets the filter number is the number of vanishing moments.
`thresh.type`	The type of thresholding function used. Currnetly only "soft" and "hard" are available. Recommended to use "soft".
`normal`	If TRUE, uses a threshold assuming the data are normally distributed. If FALSE, uses a larger threshold to reflect non-normality.
`family`	Selects the wavelet family to use. Recommended to only use the Daubechies compactly supported wavelets DaubExPhase and DaubLeAsymm.
`max.scale`	Selects the number of scales of the wavelet transform to apply thresholding to. Should be a value from 1 (fineest) to J-1 (coarsest), where T=2^J is the length of the time series. Recommended to use 2J/3 scales.
`trans.type`	The type of wavelet transform used in the procedure. Either "nondec" to use non-decimated wavelet transform and T.I. denoising (recommended), or "dec" to use the standard discrete wavelet transform.

Details

This function estimates the trend function of a locally stationary time series, by incorporating the evolutionary wavelet spterum estimate in a wavelet thresholding procedure. To use this function, first compute the spectral estimate of the time series, using the function ewspec.diff.

The function works as follows:

1. The wavelet transform of the time series is calculated.

2. The wavelet coefficeints are individually thresholded using the universal threshold sqrt(var*2*log T) and an estimate of their variance. The variance estimate is calculated using the spectral estimated, supplied by the user in the spec argument.

3. The inverse wavelet transform is applied to obtain the final estimate.

Value

A vector of length length(data) containing the trend estimate.

Author(s)

E T McGonigle

References

McGonigle, E. T., Killick, R., and Nunes, M. (2020). Modelling Time-Varying First and Second-Order Structure of Time Series via Wavelets and Differencing.

See Also

ewspec.diff, wav.trend.est

Examples

```
spec = cns(1024, filter.number = 4)
spec = putD(spec, level = 8, 1+sin(seq(from = 0, to = 2*pi,length = 1024))^2)

set.seed(120)

noise = LSWsim(spec)
sine_trend = -2*sin(seq(from=0,to=2*pi,length=1024))-
1.5*cos(seq(from=0,to=pi,length=1024))

x = sine_trend+noise

spec.est = ewspec.diff(data = x,family = "DaubExPhase", filter.number = 4,max.scale = 7)

trend.est = wav.diff.trend.est(data = x, spec = spec.est$S)

plot.ts(x, lty = 1, col = 8)
lines(sine_trend, col = 2, lwd = 2)
lines(trend.est,col = 4, lwd = 2, lty = 2)
```

`wav.trend.est` *Linear Wavelet Thresholding Trend Estimation of Second-Order Nonstationary Time Series*

Description

This function computes the linear wavelet thresholding trend estimate for a time series that may be second-order nonstationary. The function calculates the wavelet transform of the time series, sets to zero the non-boundary wavelet coffifients, then inverts the transform to give the final estimate.

Usage

```
wav.trend.est(data, filter.number = 10, family = "DaubLeAsymm",
                            scale = 1, type = "dec", boundary.handle =FALSE)
```

Arguments

data
: The time series you want to esitmate the trend function of.

filter.number
: Selects the index of the wavelet used in the estimation procedure. For Daubechies compactly supported wavelets the filter number is the number of vanishing moments.

family
: Selects the wavelet family to use. Recommended to only use the Daubechies compactly supported wavelets DaubExPhase and DaubLeAsymm.

scale
: Selects the coarsest scale of the wavelet transform to analyse to. Should be a value from 1 (coarsest) to J-1 (finest), where T=2^J is the length of the time series.

type
: The type of wavelet transform used. By default, it is "dec" which is the standard discrete wavelet transform. Can also be "nondec", which uses a non-decimated wavelet transform.

boundary.handle
: Can be TRUE or FALSE. If TRUE, the time series is boundary corrected, to get a less variable trend estimate at the boundaries of the times series. If FALSE, no boundary correction is applied.

Value

A vector of length length(data) containing the trend estimate.

Author(s)

E T McGonigle

Examples

```
##---- computes estimator of simulated linear trend time series

set.seed(1)

noise = rnorm(512)
trend = seq(from = 0, to = 5,length = 512)
x = trend+noise

trend.est = wav.trend.est(x, filter.number = 4, family = "DaubLeAsymm", boundary.handle =TRUE)

plot.ts(x, lty = 1, col = 8)
lines(trend, col = 2, lwd = 2)
lines(trend.est,col = 4, lwd = 2, lty = 2)
```

Index