Package 'WiSEBoot'

April 3, 2016

Type Package

Title Wild Scale-Enhanced Bootstrap

Version 1.4.0

Date 2016-03-31

Author Megan Heyman, Snigdhansu Chatterjee

Maintainer Megan Heyman <heyma029@umn.edu>

Description Perform the Wild Scale-Enhanced (WiSE) bootstrap. Specifically, the user may supply a single or multiple equally-spaced time series and use the WiSE bootstrap to select a wavelet-smoothed model. Conversely, a pre-selected smooth level may also be specified for the time series. Quantities such as the bootstrap sample of wavelet coefficients, smoothed bootstrap samples, and specific hypothesis testing and confidence region results of the wavelet coefficients may be obtained. Additional functions are available to the user which help format the time series before analysis. This methodology is recommended to aid in model selection and signal extraction.
Note: This package specifically uses wavelet bases in the WiSE bootstrap methodology, but the theoretical construct is much more versatile.

License GPL-2

Depends R (>= 3.1.0)

Imports wavethresh, FAdist

VignetteBuilder knitr

Suggests knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2016-04-03 16:55:59

R topics documented:

`WiSEBoot-package` *Wild Scale-Enhanced (WiSE) Bootstrap*

Description

The Wild Scale-Enhanced (WiSE) bootstrap method is implemented for models which include a wavelet signal component. Functions within this package allow the users to automatically select a wavelet smoothing level, test wavelet coefficient-specific hypotheses, and manipulate the data for usage within wavelet methodologies.

Details

Package:	WiSEBoot
Type:	Package
Version:	1.4.0
Date:	2016-03-31
License:	GPL-2

Most users of this package will need to modify their data so that it is compatible with the discrete wavelet transform. That is, the input data series for the WiSE bootstrap must be of length $T = 2^J$ for some positive integer, J. The `padVector` and `padMatrix` functions are quick and easy tools which convert the data to appropriate length. A base knowledge of wavelets is recommended before attempting to utilize this package.

The bootstrap methodology is built to automatically estimate parameters within models of the form

$$Y = \gamma_0 1 + \gamma_1 t + W\gamma + e$$

where Y is the data is the vector, linear parameters in time (t) are γ_0 and γ_1, γ are the wavelet coefficients (scaling and filter), and W is the DWT for a fixed wavelet basis. Note, in many cases of the DWT, the scaling coefficient is equivalent to γ_0, and thus, estimated there.

The `WiSEBoot` function allows the user to automatically select a threshold level for γ. Our threshold is defined as the level above which all fine wavelet coefficients are set to 0. This function also provides the WiSE bootstrap samples of the wavelet coefficients (for the selected threshold) and bootstrap samples of the linear parameters.

The `WiSEHypothesisTest` and `WiSEConfidenceRegion` functions allow the user to test or examine a specific relationship between wavelet coefficients from two data series. Specifically, given 2 data series,

$$X = \gamma_{x0}1 + \gamma_{x1}t + W\gamma_x + e_x$$

$$Y = \gamma_{y0}1 + \gamma_{y1}t + W\gamma_y + e_y$$

these functions help the user examine the viability of the relationship $\gamma_y = \alpha + \beta\gamma_x$. Note, γ_x and γ_y are defined as the non-thresholded wavelet coefficients of these series. We may obtain a p-value for the specific null hypothesis

$$H_0 : \alpha = m, \beta = n$$

(where m, n are real numbers), or visualize a confidence region in these parameters.

Author(s)

Megan Heyman, Snigdhansu Chatterjee

Maintainer: Megan Heyman <heyma029@umn.edu>

References

Errata and breaking news: http://users.stat.umn.edu/~heyma029/WiSEBoot_errata.html

Amy Braverman (NASA-JPL), Noel Cressie (Univ. of Wollongong), and Matthew Gunson (NASA-JPL) are other major contributors in projects related to developing methodology for this package.

The WiSE bootstrap methodology is defined in theoretical detail in Chatterjee, S. et al. "WiSE bootstrap for model selection" (in progress).

The WiSE bootstrap hypothesis test is implemented as an analysis tool in Braverman, A. et al. "Probabilistic Climate Model Evaluation" (in progress).

For an overview of wavelet methods using `wavethresh` in R, see "Wavelet Methods for Statistics in R," (Nason, 2008).

See Also

`wavethresh-package`

Examples

```
##User should implement a high number of bootstrap samples (R).
##  R=10 bootstrap samples is not recommended.  For demonstration only.

##Select a wavelet smooth level for signal
data("SimulatedSNR9Series")
bootObj <- WiSEBoot(SimulatedSNR9Series[,4], R=10)
bootObj$MSECriteria #check WiSEBoot selected threshold (minimum MSE) -- truth is J0=3

##Test whether \alpha=0 and \beta=1 for AIRS and IPSL Run 1 at 60E
```

```
data("CM20N20S60E")
padData <- padMatrix(CM20N20S60E)
hypTest <- WiSEHypothesisTest(padData$xPad[,1], padData$xPad[,2], J0=5, R=10,
                          XParam=padData$linearParam[,1], YParam=padData$linearParam[,2],
                               plot=TRUE)
```

`CM20N20S150W` *AIRS, IPSL, and MIROC5 Data at 150W*

Description

This matrix of data contains specific humidity observations from AIRS, 4 runs of the IPSL climate model, and 6 runs of the MIROC5 climate model. All series have been gridded according to the IPSL model grid. This data set contains 3012 observations at 150W between 20N and 20S at an altitude of 500 hPa.

Usage

```
data("CM20N20S150W")
```

Format

The format is:

num [1:3012, 1:11] 0.00334 0.0038 0.00355 0.00338 0.00383 ...

- attr(*, "dimnames")=List of 2

..$: chr [1:3012] "2002-10-01" "2002-10-02" "2002-10-03" "2002-10-04" ...

..$: chr [1:11] "AIRS" "IPSLRun1" "IPSLRun2" "IPSLRun3" ...

Details

The row and column names indicate the date and AIRS data or climate model output. Dates of this particular data set are October 1, 2002 to December 29, 2010, and observations are recorded daily.

Source

The Atmospheric Infrared Sounder (AIRS) data is described and available at `http://airs.jpl.nasa.gov/data/overview`.

The Institut Pierre Simon Laplace (IPSL) model output is described at `http://icmc.ipsl.fr/`.

The Model for Interdisciplinary Research on Climate (MIROC5) model output is described at `http://www.icesfoundation.org/Pages/ScienceItemDetails.aspx?siid=181`.

Each data source is available on a different global latitude-longitudinal grid. The data provided here was re-gridded and provided by Amy Braverman (NASA-JPL) and Matthew Gunson (NASA-JPL).

Examples

```
data(CM20N20S150W)
```

`CM20N20S60E` *AIRS, IPSL, and MIROC5 Data at 60E*

Description

This matrix of data contains specific humidity observations from AIRS, 4 runs of the IPSL climate model, and 6 runs of the MIROC5 climate model. All series have been gridded according to the IPSL model grid. This data set contains 3012 observations at 60E between 20N and 20S at an altitude of 500 hPa.

Usage

```
data("CM20N20S60E")
```

Format

The format is:

num [1:3012, 1:11] 0.00176 0.0015 0.00152 0.00182 0.00198 ...

- attr(*, "dimnames")=List of 2

..$: chr [1:3012] "2002-10-01" "2002-10-02" "2002-10-03" "2002-10-04" ...

..$: chr [1:11] "AIRS" "IPSLRun1" "IPSLRun2" "IPSLRun3" ...

Details

The row and column names indicate the date and AIRS data or climate model output. Dates of this particular data set are October 1, 2002 to December 29, 2010, and observations are recorded daily.

Source

The Atmospheric Infrared Sounder (AIRS) data is described and available at `http://airs.jpl.nasa.gov/data/overview`.

The Institut Pierre Simon Laplace (IPSL) model output is described at `http://icmc.ipsl.fr/`.

The Model for Interdisciplinary Research on Climate (MIROC5) model output is described at `http://www.icesfoundation.org/Pages/ScienceItemDetails.aspx?siid=181`.

Each data source is available on a different global latitude-longitudinal grid. The data provided here was re-gridded and provided by Amy Braverman (NASA-JPL) and Matthew Gunson (NASA-JPL).

Examples

```
data(CM20N20S60E)
```

deSeasonalize — *De-seasonalize daily, monthly, or data series with IDs*

Description

Generally, this function performs data standardization by an ID. It is useful for data pre-processing, by removing daily, monthly, or other periodic means which are not of interest. Two quick methods are available: removing the mean and standardization. Note, this is not the optimal method in the statistical literature for de-seasonalization.

Usage

```
deSeasonalize(dates, X, type = "daily", method = "deMean")
```

Arguments

`dates`	vector of dates or IDs for the data, `X`. This vector must have the same length as `X` (or the same number of rows as `X`). A recognized R date format is required unless the `method="custom"` option is called. Missing values are not allowed. More than 1 replicate of the ID is recommended, as this is the criteria to remove the mean/standardize.
`X`	vector or matrix of all data to be de-seasonalized. Missing values are not allowed and the data should be numeric. The length, or number of rows, of `X` should match the length of `dates`. Thus, if `X` is a matrix, the columns of `X` contain the individual data series.
`type`	how often to de-seasonalize. Allowed values are `"daily"`, `"monthly"`, and `"custom"`. The `"daily"` or `"monthly"` de-seasonalization options require that the `dates` vector be of a recognized R date format. The `"custom"` option allows for de-seasonalization at other rates (e.g. quarterly, hourly, etc.), or, more generally, standardization by an ID.
`method`	how de-seasonalization is implemented. Allowed values are `"deMean"` and `"standardize"`. The `"deMean"` method removes the mean by `dates`. The `"standardize"` method standardizes (i.e. removes the mean and divides by the standard deviation) by `dates`.

Details

For a supplied matrix, `X`, the same `dates` – ID criteria – is used for de-seasonalization in each column. That is, each column of `X` is de-seasonalized by `dates`.

This is not an optimal de-seasonalization methodology. For users interested in a more robust method, please see the '`deseasonalize`' R package.

Value

a vector or matrix of the same dimension as `X` which has been de-seasonalized appropriately.

Author(s)

Megan Heyman

See Also

R package 'deseasonalize', `http://cran.r-project.org/package=deseasonalize`

Examples

```
ID <- as.Date(c("2014-05-14", "2013-06-20", "2013-05-14", "2012-06-20",
                "1999-09-09", "1998-09-08", "1998-09-09", "1982-05-14",
                "2000-09-08"))
someData <- seq(1, 9)

deSeasonalize(dates=ID, X=someData, type="daily", method="deMean")
deSeasonalize(dates=ID, X=someData, type="monthly", method="standardize")
```

`padMatrix` *Increase data length to the closest power of 2.*

Description

To use the WiSE bootstrap methodology in this package, data must be of length $T = 2^J$ for some positive integer, J. This function increases the length of data to achieve the particular length requirement. Generally, this function is useful for data pre-processing.

Usage

```
padMatrix(X, by.row = TRUE, type = "reflect", pad.direction = "both",
          replaceLinearTrend = FALSE)
```

Arguments

`X`	a matrix of data. This must be numeric and non-missing.
`by.row`	logical indicator of observation location. If `by.row=TRUE`, the matrix contains observations in the rows and each column represents a different data series (padding data by column). If `by.row=FALSE`, the matrix contains observations in the columns and each row represents a different data series (padding data by row).
`type`	how to increase the data length. Allowed values are `"reflect"`, `"periodic"`, or `"mean"`. The `"reflect"` option repeats values proceeding the end/beginning of the series (ex: `12345 --> 12345432`). The `"periodic"` option repeats the series in order (ex: `12345 --> 12345123`). The `"mean"` option repeats the series mean (ex: `12345 --> 12345333`).
`pad.direction`	where to add the data padding. Allowed values are `"both"`, `"front"`, or `"rear"`. The `"both"` option pads data on both sides of the series (ex: `12345 --> 31234533`). The `"front"` option pads data on the beginning of the series (ex: `12345 --> 33312345`). The `"rear"` option pads data on the end of the series (ex: `12345 --> 12345333`).

`replaceLinearTrend`	logical. If `TRUE`, the estimated linear trend is replaced in the returned data. If `FALSE`, the estimated trend is not replaced in the returned data. See Details for more information.

Details

If the data supplied is already a power of 2, this function will just return the original or de-trended data.

If the data supplied is of length, t, the padded data returned will be of length $T = 2^{ceiling(log(t, base=2))}$.

The data length $T = 2^J$ for a positive integer, J, requirement is associated with the discrete wavelet transform. Although methodology exists in the wavelet literature which allows for data series of any length, this methodology does not align with the theory behind WiSE bootstrap.

The `replaceLinearTrend` option allows the user to control whether linear trend appears in the padded data. The linear trend (by data index) is estimated using least squares for each data series. This trend is removed before padding the data. The estimated trend may or may not be replaced to the padded data. The linear trend consists of the data intercept and slope (by index).

Value

`xPad`	a matrix of padded data. This matrix contains the same number of supplied data series, but the data series will be of length T instead of t (see Details).
`origSeriesIndex`	a vector of 2 indices. These indicate where the original data is in the padded series. Note, the `xPad` will not exactly match the original data between these indices, since the linear trend has been estimated (and possibly replaced).
`linearParam`	a matrix with 2 rows. The first row is the least squares estimated intercept and the second is the least squares estimated slope (by index) from the data originally supplied. The columns correspond to the individual data series, in order of `X`.
`by.row`	same as supplied to the function.

Author(s)

Megan Heyman

See Also

`padVector`, `wavethresh-package`

Examples

```
someData <- matrix(seq(1,5^2)+rnorm(25), nrow=7)

padMatrix(someData)
padMatrix(someData, type="mean", pad.direction="rear")
padMatrix(someData, type="periodic", pad.direction="front")
```

`padVector` *Increase data length to the closest power of 2.*

Description

To use the WiSE bootstrap methodology in this package, data must be of length $T = 2^J$ for some positive integer, J. This function increases the length of data to achieve the particular length requirement. Generally, this function is useful for data pre-processing.

Usage

```
padVector(X, type = "reflect", pad.direction = "both",
          replaceLinearTrend = FALSE)
```

Arguments

`X`	a vector of data. This must be numeric and non-missing.
`type`	how to increase the data length. Allowed values are `"reflect"`, `"periodic"`, or `"mean"`. The `"reflect"` option repeats values proceeding the end/beginning of the series (ex: `12345 --> 12345432`). The `"periodic"` option repeats the series in order (ex: `12345 --> 12345123`). The `"mean"` option repeats the series mean (ex: `12345 --> 12345333`).
`pad.direction`	where to add the data padding. Allowed values are `"both"`, `"front"`, or `"rear"`. The `"both"` option pads data on both sides of the series (ex: `12345 --> 31234533`). The `"front"` option pads data on the beginning of the series (ex: `12345 --> 33312345`). The `"rear"` option pads data on the end of the series (ex: `12345 --> 12345333`).
`replaceLinearTrend`	logical. If `TRUE`, the estimated linear trend is replaced in the returned data. If `FALSE`, the estimated trend is not replaced in the returned data. See Details for more information.

Details

If the data supplied is already a power of 2, this function will just return the original or de-trended data.

If the data supplied is of length, t, the padded data returned will be of length $T = 2^{ceiling(log(t,base=2))}$.

The data length $T = 2^J$ for a positive integer, J, requirement is associated with the discrete wavelet transform. Although methodology exists in the wavelet literature which allows for data series of any length, this methodology does not align with the theory behind WiSE bootstrap.

The `replaceLinearTrend` option allows the user to control whether linear trend appears in the padded data. The linear trend (by data index) is estimated using least squares for the data series. This trend is removed before padding the data. The estimated trend may or may not be replaced to the padded data. The linear trend consists of the data intercept and slope (by index).

Value

`xPad`	a vector of padded data. This vector will be of length T instead of t (see Details).
`origSeriesIndex`	a vector of 2 indices. These indicate where the original data is in the padded series. Note, the `xPad` will not exactly match the original data between these indices, since the linear trend has been estimated (and possibly replaced).
`linearParam`	a vector of 2 numbers. The first is the least squares estimated intercept and the second is the least squares estimated slope (by index) from the data originally supplied.

Author(s)

Megan Heyman

See Also

`padMatrix`, `wavethresh-package`

Examples

```
someData <- seq(1,9)+ rnorm(9)

padVector(someData)
padVector(someData, type="mean", pad.direction="rear")
padVector(someData, type="periodic", pad.direction="front")
```

`retrieveBootstrapSample` *Construct the bootstrap data series from wavelet coefficients*

Description

Use the wavelet coefficients from the selected WiSE bootstrap model to construct smooth bootstrap series in the original time/space domain. This function may be of little use and creates large array objects within R. The wavelet coefficients are a preferable representation of the data, as they are sparse and contain the signal information.

Usage

```
retrieveBootstrapSample(WiSEObj)
```

Arguments

`WiSEObj`	an object obtained from `WiSEBoot`. The `"periodic"` boundary condition for wavelets is required.

Details

The wavethresh package is used to perform the inverse wavelet decomposition of each bootstrap sample of wavelet coefficients.

The bootstrap series will be smoothed to the selected threshold level, $J0 = j$ (see WiSEBoot).

Value

BootSample	an array. The rows contain each bootstrap sample – dimension is the R supplied to the original WiSEBoot call. The columns contain the observations in the original time/space domain – dimension is the length of the data series, T. The 3rd dimension corresponds to the data series – If the user supplied a matrix of more than 1 data series, a bootstrap sample is generated for each series, in the order of the data series supplied.

Author(s)

Megan Heyman

References

For an overview of the wavelet methodology used in wavethresh, see "Wavelet Methods for Statistics in R," (Nason, 2008).

See Also

WiSEBoot, wavethresh-package

Examples

```
someData <- rnorm(2^5)

##Bootstrap sample of size 10 is not recommended. For demonstration only.
bootInfo <- WiSEBoot(someData, R=10, J0=2)
bootSeries <- retrieveBootstrapSample(bootInfo)$BootSample

bootSeries[1, , 1] #this is the first bootstrap series
```

SimulatedSmoothSeries *Simulated Wavelet-Smoothed Series*

Description

A matrix containing some simulated, smooth data series. The smooth series are contained in the columns, and the data observations in the rows. An inverse wavelet decomposition of the series exactly yields thresholded coefficients at the specified level ($J0 = j$).

Usage

```
data("SimulatedSmoothSeries")
```

Format

The format is:

num [1:1024, 1:9] -0.00934 -0.00934 -0.00935 -0.00935 -0.00935 ...

- attr(*, "dimnames")=List of 2

..$: NULL

..$: chr [1:9] "J0.0" "J0.1" "J0.2" "J0.3" ...

Details

The columns contain series created by different wavelet coefficient threshold levels. Smoothed series are available for thresholds of $J0$ in $\{0, 1, 2, 3, 4, 5, 6, 7, 8\}$. The rows are the data observations. Thus, each smooth series is of length $2^{10} = 1024$.

The names of each column indicate the threshold ($J0$). For example, the 3rd column, named `J0.2`, has a threshold of $J0 = 2$, and thus 0-valued wavelet coefficients for all mother wavelet coefficients finer than level 2.

These smooth series were generated using `'wd'` and `'wr'` with the `family="DaubLeAsymm"`, `filter.number=8`, `bc="periodic"` options in the `'wavethresh'` package.

References

The user may want these smooth series to aid in simulations with this package.

Also see `wavethresh-package`

`SimulatedSNR15Series` *Simulated Wavelet Signals with SNR=15*

Description

A matrix containing simulated signals with noise added such that the signal-to-noise ratio (SNR) is 15. If the signal vector is called Y, and Y is of length 2^J, we have defined the SNR within the $J0 = j$ threshold as

$$SNR = (1/2^J)((< Y, Y > /(2^j - 1))/(\sigma^2/(2^J - 2^j - 1)))$$

where σ^2 is the variance of the noise. These series are obtained by adding white noise to the smooth data in `SimulatedSmoothSeries`. Each column contains a wavelet coefficient threshold level and rows contain observations.

Usage

```
data("SimulatedSNR15Series")
```

Format

The format is:

num [1:1024, 1:9] -0.01579 -0.00478 -0.00508 -0.01277 -0.01427 ...

- attr(*, "dimnames")=List of 2

..$: NULL

..$: chr [1:9] "J0.0" "J0.1" "J0.2" "J0.3" ...

Details

The columns contain noisy series with signals in different wavelet coefficient threshold levels. Series are available for signal thresholds of $J0$ in $\{0, 1, 2, 3, 4, 5, 6, 7, 8\}$. The rows are the data observations. Thus, each smooth series is of length $2^{10} = 1024$.

The names of each column indicate the threshold ($J0$) in the smooth series. For example, the 3rd column, named `J0.2`, has a threshold of $J0 = 2$ in the signal, and thus 0-valued wavelet coefficients for all mother wavelet coefficients finer than level 2 in the signal. Notice, the white noise added to the signal creates non-zero coefficients above the threshold.

The original smooth series were generated using `'wd'` and `'wr'` with the `family="DaubLeAsymm"`, `filter.number=8`, `bc="periodic"` options in the `'wavethresh'` package.

References

wavethresh-package, SimulatedSmoothSeries

Examples

```
data(SimulatedSNR15Series)

##See if WiSEBoot selects the correct threshold for this data (J0=3)
## R=10 bootstrap samples is not recommended.  For demonstration only.
bootObj <- WiSEBoot(SimulatedSNR15Series[,4], R=10)
bootObj$MSECriteria

##Look at the noisy data compared to the true smooth
data(SimulatedSmoothSeries)
plot(seq(1, 2^10), SimulatedSNR15Series[ , 6], main="Threshold of J0=5",
     col="lightgray", xlab="Time", ylab="Observations", type="l")
lines(seq(1, 2^10), SimulatedSmoothSeries[ ,6], col="red", lwd=2)
```

`SimulatedSNR25Series` *Simulated Wavelet Signals with SNR=25*

Description

A matrix containing simulated signals with noise added such that the signal-to-noise ratio (SNR) is 25. If the signal vector is called Y, and Y is of length 2^J, we have defined the SNR within the $J0 = j$ threshold as

$$SNR = (1/2^J)((< Y, Y > /(2^j - 1))/(\sigma^2/(2^J - 2^j - 1)))$$

where σ^2 is the variance of the noise. These series are obtained by adding white noise to the smooth data in `SimulatedSmoothSeries`. Each column contains a wavelet coefficient threshold level and rows contain observations.

Usage

```
data("SimulatedSNR25Series")
```

Format

The format is:

num [1:1024, 1:9] -0.01014 -0.01667 -0.0086 -0.01215 -0.00873 ...

- attr(*, "dimnames")=List of 2

..$: NULL

..$: chr [1:9] "J0.0" "J0.1" "J0.2" "J0.3" ...

Details

The columns contain noisy series with signals in different wavelet coefficient threshold levels. Series are available for signal thresholds of $J0$ in $\{0, 1, 2, 3, 4, 5, 6, 7, 8\}$. The rows are the data observations. Thus, each smooth series is of length $2^{10} = 1024$.

The names of each column indicate the threshold ($J0$) in the smooth series. For example, the 3rd column, named `J0.2`, has a threshold of $J0 = 2$ in the signal, and thus 0-valued wavelet coefficients for all mother wavelet coefficients finer than level 2 in the signal. Notice, the white noise added to the signal creates non-zero coefficients above the threshold.

The original smooth series were generated using `'wd'` and `'wr'` with the `family="DaubLeAsymm"`, `filter.number=8`, `bc="periodic"` options in the `'wavethresh'` package.

References

`wavethresh-package`, `SimulatedSmoothSeries`

Examples

```
data(SimulatedSNR25Series)

##See if WiSEBoot selects the correct threshold for this data (J0=3)
## R=10 bootstrap samples is not recommended.  For demonstration only.
bootObj <- WiSEBoot(SimulatedSNR25Series[,4], R=10)
bootObj$MSECriteria
```

```
##Look at the noisy data compared to the true smooth
data(SimulatedSmoothSeries)
plot(seq(1, 2^10), SimulatedSNR25Series[ , 6], main="Threshold of J0=5",
     col="lightgray", xlab="Time", ylab="Observations", type="l")
lines(seq(1, 2^10), SimulatedSmoothSeries[ ,6], col="red", lwd=2)
```

SimulatedSNR5Series *Simulated Wavelet Series with SNR=5*

Description

A matrix containing simulated signals with noise added such that the signal-to-noise ratio (SNR) is 5. If the signal vector is called Y, and Y is of length 2^J, we have defined the SNR within the $J0 = j$ threshold as

$$SNR = (1/2^J)((< Y, Y > /(2^j - 1))/(\sigma^2/(2^J - 2^j - 1)))$$

where σ^2 is the variance of the noise. These series are obtained by adding white noise to the smooth data in `SimulatedSmoothSeries`. Each column contains a wavelet coefficient threshold level and rows contain observations.

Usage

```
data("SimulatedSNR5Series")
```

Format

The format is:

num [1:1024, 1:9] -0.01354 -0.01315 -0.00491 -0.01808 0.00472 ...

- attr(*, "dimnames")=List of 2

..$: NULL

..$: chr [1:9] "J0.0" "J0.1" "J0.2" "J0.3" ...

Details

The columns contain noisy series with signals in different wavelet coefficient threshold levels. Series are available for signal thresholds of $J0$ in {0, 1, 2, 3, 4, 5, 6, 7, 8}. The rows are the data observations. Thus, each smooth series is of length $2^{10} = 1024$.

The names of each column indicate the threshold ($J0$) in the smooth series. For example, the 3rd column, named `J0.2`, has a threshold of $J0 = 2$ in the signal, and thus 0-valued wavelet coefficients for all mother wavelet coefficients finer than level 2 in the signal. Notice, the white noise added to the signal creates non-zero coefficients above the threshold.

The original smooth series were generated using `'wd'` and `'wr'` with the `family="DaubLeAsymm"`, `filter.number=8`, `bc="periodic"` options in the `'wavethresh'` package.

References

wavethresh-package, SimulatedSmoothSeries

Examples

```
data(SimulatedSNR5Series)

##See if WiSEBoot selects the correct threshold for this data (J0=3)
## R=10 bootstrap samples is not recommended.  For demonstration only.
bootObj <- WiSEBoot(SimulatedSNR5Series[,4], R=10)
bootObj$MSECriteria

##Look at the noisy data compared to the true smooth
data(SimulatedSmoothSeries)
plot(seq(1, 2^10), SimulatedSNR5Series[ , 6], main="Threshold of J0=5",
     col="lightgray", xlab="Time", ylab="Observations", type="l")
lines(seq(1, 2^10), SimulatedSmoothSeries[ ,6], col="red", lwd=2)
```

SimulatedSNR9Series *Simulated Wavelet Signals with SNR=9*

Description

A matrix containing simulated signals with noise added such that the signal-to-noise ratio (SNR) is 9. If the signal vector is called Y, and Y is of length 2^J, we have defined the SNR within the $J0 = j$ threshold as

$$SNR = (1/2^J)((< Y, Y > /(2^j - 1))/(\sigma^2/(2^J - 2^j - 1)))$$

where σ^2 is the variance of the noise. These series are obtained by adding white noise to the smooth data in `SimulatedSmoothSeries`. Each column contains a wavelet coefficient threshold level and rows contain observations.

Usage

```
data("SimulatedSNR9Series")
```

Format

The format is:

num [1:1024, 1:9] -0.01879 -0.01635 -0.01096 -0.00814 -0.00423 ...

- attr(*, "dimnames")=List of 2

..$: NULL

..$: chr [1:9] "J0.0" "J0.1" "J0.2" "J0.3" ...

Details

The columns contain noisy series with signals in different wavelet coefficient threshold levels. Series are available for signal thresholds of $J0$ in $\{0, 1, 2, 3, 4, 5, 6, 7, 8\}$. The rows are the data observations. Thus, each smooth series is of length $2^{10} = 1024$.

The names of each column indicate the threshold ($J0$) in the smooth series. For example, the 3rd column, named `J0.2`, has a threshold of $J0 = 2$ in the signal, and thus 0-valued wavelet coefficients for all mother wavelet coefficients finer than level 2 in the signal. Notice, the white noise added to the signal creates non-zero coefficients above the threshold.

The original smooth series were generated using `'wd'` and `'wr'` with the `family="DaubLeAsymm"`, `filter.number=8`, `bc="periodic"` options in the `'wavethresh'` package.

References

`wavethresh-package`, `SimulatedSmoothSeries`

Examples

```
data(SimulatedSNR9Series)

##See if WiSEBoot selects the correct threshold for this data (J0=3)
## R=10 bootstrap samples is not recommended.  For demonstration only.
bootObj <- WiSEBoot(SimulatedSNR9Series[,4], R=10)
bootObj$MSECriteria

##Look at the noisy data compared to the true smooth
data(SimulatedSmoothSeries)
plot(seq(1, 2^10), SimulatedSNR9Series[ , 6], main="Threshold of J0=5",
     col="lightgray", xlab="Time", ylab="Observations", type="l")
lines(seq(1, 2^10), SimulatedSmoothSeries[ ,6], col="red", lwd=2)
```

`smoothTimeSeries` *Threshold Wavelet Coefficients to Create Smooth Time Series*

Description

This function takes a vector of an equally-spaced time series which is of length $T = 2^J$ for a positive integer, J. The series is thresholded at all wavelet coefficient levels and a matrix is returned which contains each of the wavelet-smoothed series. Optionally, the user may plot none, one, or all of the wavelet smoothed series compared to the original input data.

This function is most useful for visualization of the data and primarily used within the WiSE bootstrap methodology.

Usage

```
smoothTimeSeries(X, wavFam = "DaubLeAsymm", wavFil = 8, wavBC = "periodic",
                 plotLevels = "none", ...)
```

Arguments

`X`	a vector containing an equally-spaced data series. The vector must be of length $T = 2^J$ for some integer, $J > 3$, contain no missing values, and be numeric. If the data is a single-row or column matrix, it must be input to this function as a vector.
`wavFam`	wavelet family. Allowed values are `"DaubLeAsymm"` and `"DaubExPhase"` – Daubechies Least Asymmetric and Daubechies Extremal Phase. This is the `family` used within the `wavethresh` package.
`wavFil`	wavelet filter number. Allowed values are integers between 4 and 10 when `wavFam="DaubLeAsymm"` or integers between 1 and 10 when `wavFam="DaubExPhase"`. These correspond to the number of vanishing moments of the wavelet. This is the `filter.number` used within the `wavethresh` package.
`wavBC`	wavelet boundary condition. Allowed values are `"periodic"` and `"symmetric"`. This is the `bc` used within the `wavethresh` package.
`plotLevels`	plotting option. Allowed values are `"none"`, `"all"`, or an integer between 0 and $J - 1$.
`...`	additional graphical arguments. See `plot`, `plot.default`.

Details

To smooth the series to the $J0 = j$ level, all wavelet coefficients at levels $j + 1$ and finer are set to 0. The reverse wavelet transformation is performed upon the thresholded coefficients to obtain each smooth data series. This method produces smooth data series for $J0 + 1$ in $\{0, 1, ..., J - 1\}$. Note, when $J0 + 1 = 0$, this indicates that ALL filter wavelet coefficients have been set to 0.

If `"all"` plots are requested, all possible smooth series are plotted against the original input data within one graphical device. This is recommended for users trying to visualize the level of smoothing needed in their data. Note, $J0 + 1$ corresponds to the first fine level of coefficients which is set entirely to 0.

The wavelet options are only those allowed within the `wavethresh` package. Please see the documentation for `wavethresh` for further explanation of these quantities.

Value

a matrix containing the smooth series. The column names represent the smooth-level of the data series. The first column contains the original input data. The second column contains the wavelet smooth which sets only the finest level of coefficients to 0 (i.e. $J0 + 1 = J - 1$). The last column contains the smooth which sets all mother wavelet coefficients to 0 (i.e. $J0 + 1 = 0$). Notice, the column names should help indicate the smooth level. The matrix rows contain the ordered observations.

Author(s)

Megan Heyman

See Also

To obtain data series of the appropriate length: `padVector`.

Documentation regarding the wavelet transform: `wavethresh-package`

Examples

```
##Visualize data smoothing on the AIRS 60E data
data(CM20N20S60E)
AIRS <- as.vector(CM20N20S60E[ ,1])
padAIRS <- padVector(AIRS)$xPad
smoothAIRS <- smoothTimeSeries(padAIRS, plotLevels="all")
```

`WiSEBoot` *Wild Scale-Enhanced (WiSE) Bootstrap for Model Selection*

Description

Perform the WiSE bootstrap to estimate parameters within models of the form

$$Y = \gamma_0 1 + \gamma_1 t + W\gamma + e$$

Automatically select a threshold level for γ, or the user may specify the threshold. This function also provides the WiSE bootstrap samples of the wavelet coefficients (for the selected threshold) and bootstrap samples of the linear parameters (γ_0, γ_1).

Usage

```
WiSEBoot(X, R=100, XParam = NA, TauSq = "log", bootDistn = "normal", by.row = FALSE,
         J0 = NA, wavFam = "DaubLeAsymm", wavFil = 8, wavBC = "periodic")
```

Arguments

`X`	a matrix or vector of equally-spaced data. All entries must be non-missing and numeric. If a vector is supplied, the length must be $T = 2^J$ where J is a positive integer. If a matrix is supplied, each data series must be of length T.
`R`	number of bootstrap samples. Allowed value is a positive integer. Default is 100.
`XParam`	vector or matrix of linear trend parameters for the data series. If a vector is supplied in `X`, this should be a vector of length 2. If a matrix is supplied in `X`, this should be a matrix with 2 rows and an equal of columns (`dim(X)[2]=dim(XParam)[2]`). If `NA`, `WiSEBoot` will automatically estimate linear parameters via least squares. This quantity may be supplied by the `linearParam` return argument of `padVector` or `padMatrix`. See Details below.
`TauSq`	scale parameter for the bootstrap. Allowed values are `"log"`, `"log10"`, `"sqrt"`, `"1"`, or `"2/5"`. The scale parameter is related to the length of the data series. For example, `"log"` implies a value of the scale parameter, τ, of $\sqrt{log(T)}$. The value of `"1"` creates an equivalent situation to wild bootstrap.

`bootDistn`	the distribution for the bootstrap. Allowed values are `"normal"`, `"uniform"`, `"laplace"`, `"lognormal"`, `"gumbel"`, `"exponential"`, `"t5"`, `"t8"`, and `"t14"`. This draws iid random samples from the specified distribution for the wild bootstrap where the random variables have mean 0 and variance 1. For example, `"t5"` is Student's t-distribution with 5 degrees of freedom.
`by.row`	logical indicator of observation location. If `TRUE`, the observations are by row and the columns contain different data series. If `FALSE`, the rows contain different data series and the observations are by column.
`J0`	wavelet filter coefficient threshold. Allowed values are `NA` and any integer between 0 and $J-2$ (when the data series is of length $T = 2^J$). If a specific integer is given, all wavelet coefficients at levels finer than `J0` are set to 0. If `NA`, the WiSEBoot creates bootstrap samples for all thresholds between 0 and $J-2$. The selected threshold minimizes the mean of the MSE.
`wavFam`	wavelet family. Allowed values are `"DaubLeAsymm"` and `"DaubExPhase"` – Daubechies Least Asymmetric and Daubechies Extremal Phase. This is the `family` used within the `wavethresh` package.
`wavFil`	wavelet filter number. Allowed values are integers between 4 and 10 when `wavFam="DaubLeAsymm"` or integers between 1 and 10 when `wavFam="DaubExPhase"`. These correspond to the number of vanishing moments of the wavelet. This is the `filter.number` used within the `wavethresh` package.
`wavBC`	wavelet boundary condition. Allowed values are `"periodic"` and `"symmetric"`. This is the `bc` used within the `wavethresh` package.

Details

The assumed model is

$$Y = \gamma_0 1 + \gamma_1 t + W\gamma + e$$

where Y is the data vector, linear parameters in time (t) are γ_0 and γ_1, γ are the wavelet coefficients (scaling and filter), and W is the DWT for a fixed wavelet basis. Note, in many cases of the DWT, the scaling coefficient is equivalent to γ_0, and thus, estimated there.

This model requires estimation of linear terms γ_0 and γ_1. It is recommended, if the data is padded to a length $T = 2^J$ using `padVector` or `padMatrix`, to supply the `linearParam` estimates and call `replaceLinearTrend=FALSE`. If `XParam` is `NA`, the `WiSEBoot` function will estimate γ_0 and γ_1 from the supplied data using least squares.

`J0` sets the threshold within the wavelet coefficients. Our threshold is defined as the level above which all fine wavelet coefficients are set to 0.

For a single data series, Y, the WiSE bootstrap sample is obtained by

1. Find estimates of γ_0 and γ_1: $g0$ and $g1$. If supplied, this is `linearParam`.

2. Estimate all levels of wavelet coefficients, γ, using the residuals $r = Y - g01 - g1t$. Call these estimated coefficients g.

3. For a set threshold, $J0 = j$, set all coefficients in g finer than j to 0. Call this thresholded set of coefficients g_j Perform the inverse wavelet transform with g_j. This smooth series may be called $rSmooth$.

4. Calculate the wavelet residuals using $rWave = r - rSmooth$.

5. A single bootstrap sample is defined as $Y^* = g01 + g1t + Wg_j + \tau N(0,1)rWave$. Y^* is used to obtain estimates for the un-thresholded wavelet coefficients and linear parameters.

Value

`MSECriteria`	matrix with 2 columns. The first column contains integer values corresponding to various $J0+1$. The second column contains the mean of the MSE. The MSE is computed using the estimated (smooth) bootstrap sample and the original data. The selected model minimizes the mean of MSE.
`BootIntercept`	matrix of `R` rows. Each row corresponds to a single bootstrap sample estimate of γ_0. The number of columns in the matrix corresponds to the number of data series supplied to the function (in `X`). The order of the data series supplied matches to the order of the columns of `BootIntercept`.
`BootSlope`	matrix of `R` rows. Each row corresponds to a single bootstrap sample estimate of γ_1. The number of columns in the matrix corresponds to the number of data series supplied to the function (in `X`). The order of the data series supplied matches to the order of the columns of `BootSlope`.
`BootWavelet`	array of bootstrap estimates of the wavelet coefficients. The first dimension is `R` (bootstrap sample). The second dimension is the $T = 2^J$ (data series length). The order of wavelet coefficients in the second dimension is: scaling level 0, filter level 0 (coarsest), filter level 1, ..., filter level $J-1$ (finest). The third dimension is the number of data series supplied. This array does not contain any boundary coefficients generated using the `wavBC="symmetric"` option.
`DataWavelet`	matrix of wavelet coefficients from the data. This matrix only contains the coefficients from the selected model (fine level coefficients are set to 0). The number of rows is $T = 2^J$ and the order of coefficients within these rows matches the order of coefficients in the columns of `BootWavelet`. Thus, the first row contains the scaling coefficient, the second row contains the filter level 0 coefficient, etc. The number of columns matches the number of data series supplied, and are ordered as in `X`.
`XParam`	matrix of linear slope and intercept in time from the data. If supplied, same as user-specified. Otherwise, estimated using least squares.
`wavFam, wavFil, wavBC, TauSq, BootDistn, by.row`	Same as supplied to the function.

Author(s)

Megan Heyman

References

The WiSE bootstrap methodology is defined in theoretical detail in Chatterjee, S. et al. "WiSE bootstrap for model selection" (in progress).

See Also

`padVector`, `padMatrix`, `wavethresh-package`

Examples

```
##R=10 bootstrap samples is not recommended.  For demonstration only.

##bootstrap one of the simulated series, threshold level 4 (not the truth)
data(SimulatedSNR15Series)
bootObj <- WiSEBoot(SimulatedSNR15Series[, 3], R=10, J0=4)

#boxplot of the bootstrap intercept and slope estimates (both 0 in truth)
par(mfrow=c(1,2))
boxplot(bootObj$BootIntercept); boxplot(bootObj$BootSlope)

#boxplot of the bootstrap wavelet coefficient estimates, level 1
par(mfrow=c(1,2))
boxplot(bootObj$BootWavelet[ , 3, 1]); boxplot(bootObj$BootWavelet[ , 4, 1])


##See what smooth level the bootstrap chooses (truth is J0=2)
bootObj2 <- WiSEBoot(SimulatedSNR15Series[ ,3], R=10)
bootObj2$MSECriteria
```

`WiSEConfidenceRegion` *WiSE Wavelet Coefficients: Linear Confidence Region*

Description

Calculate the WiSE bootstrap sample of the linear parameters describing a set of wavelet coefficients from two WiSEBoot objects. See Details for a precise description.

Usage

```
WiSEConfidenceRegion(X, Y, plot = TRUE, ...)
```

Arguments

`X`	`WiSEBoot` object for a single data series. The data series length from this `WiSEBoot` object should match the data series length for the object in `Y`. The selected threshold within this object should match the selected threshold for the object in `Y`. The number of bootstrap samples within this object should match the number of samples in `Y`. It is recommended that the wavelet settings match between `X` and `Y`.
`Y`	`WiSEBoot` object for a single data series. The data series length from this `WiSEBoot` object should match the data series length for the object in `X`. The selected threshold within this object should match the selected threshold for the object in `X`. The number of bootstrap samples within this object should match the number of samples in `X`. It is recommended that the wavelet settings match between `X` and `Y`.
`plot`	logical. If `TRUE`, a plot of the bootstrap sample of linear parameters in the wavelet coefficients is generated.
`...`	additional graphical arguments. See `plot`, `plot.default`.

Details

Given 2 vectors of equally-spaced data of length $T = 2^J$ for a positive integer, J, we assume the following models:

$$X = \gamma_{x0} 1 + \gamma_{x1} t + W\gamma_x + e_x$$

$$Y = \gamma_{y0} 1 + \gamma_{y1} t + W\gamma_y + e_y$$

The WiSE bootstrap is performed on each data series with the same threshold level, $J0 = j$, using `WiSEBoot`. Bootstrap estimates of γ_x and γ_y are obtained from these `WiSEBoot` objects, called g^*_{xj} and g^*_{yj}. The estimates of γ_x and γ_y from the input data may be noted as g_{xj} and g_{yj}.

This function allows the user to examine a linear relationship between the two sets of wavelet coefficients:

$$\gamma_y = \alpha 1 + \beta \gamma_x$$

Thus, we use g^*_{xj} and g^*_{yj} to obtain estimates of α and β, called a^* and b^*. Likewise, the original coefficients from the data, g_{xj} and g_{yj}, yield estimates, a and b. The bootstrap parameter estimates allow the user to visualize the distribution of the linear parameters, α and β. Generally, these parameters give us an idea about the relationship between the two data series signals.

Currently, this function does not calculate a (1-α)% 2-dimensional region in the parameters. It allows a qualitative visualization of the bootstrap sample distribution for these linear parameters. Asymptotically, the linear parameters follow a multivariate normal distribution when assumptions are met.

Note, the slope and intercept parameters here (α and β) are different from the linear parameters in time within our data series ($\gamma_{x0}, \gamma_{y0}, \gamma_{x1}, \gamma_{y1}$). The `BootSlope` and `BootIntercept` output from `WiSEBoot` represent the linear parameters in time within the data series.

Value

`dataIntercept`	point estimate of intercept of the wavelet coefficients from the data. Using the notation in the Details section, a.
`dataSlope`	point estimate of slope of the wavelet coefficients from the data. Using the notation in the Details section, b.
`bootIntercept`	bootstrap estimates of the intercept of the wavelet coefficients. Using the notation in the Details section, a^*. This is a vector.
`bootSlope`	bootstrap estimates of the slope of the wavelet coefficients. Using the notation in the Details section, b^*. This is a vector.

Author(s)

Megan Heyman

References

The WiSE bootstrap hypothesis test is implemented as an analysis tool in Braverman, A. et al. "Probabilistic Climate Model Evaluation" (in progress). This is the corresponding confidence region to the calculations presented there.

See Also

WiSEBoot, wavethresh-package

Examples

```
## R=10 bootstrap samples is not recommended. For demonstration only.


###Example with random data
x <- rnorm(2^8)
y <- x + rnorm(2^8, sd=0.001) #y has similar structure to x
xWise <- WiSEBoot(x, R=10, J0=4)
yWise <- WiSEBoot(y, R=10, J0=4)
xyConf <- WiSEConfidenceRegion(xWise, yWise) #does the region contain (0, 1)?


###Example with AIRS and IPSL data
data(CM20N20S60E)
padCM <- padMatrix(CM20N20S60E)  #pad data so we can use wavelet methodology

AIRS <- WiSEBoot(padCM$xPad[,1], R=10, J0=5, XParam=padCM$linearParam[,1])
IPSL1 <- WiSEBoot(padCM$xPad[,2], R=10, J0=5, XParam=padCM$linearParam[,2])

AIRS_IPSL1Conf <- WiSEConfidenceRegion(AIRS, IPSL1) #how are these signals related?
```

WiSEHypothesisTest *WiSE Wavelet Coefficients: Linear Hypothesis Test*

Description

Calculate the p-value for a hypothesis test regarding a linear relationship between wavelet coefficients from two data series. See Details for a precise description.

Usage

```
WiSEHypothesisTest(X, Y, J0, R=100, popParam = c(0, 1), XParam = c(NA, NA),
                     YParam = c(NA, NA), TauSq = "log", bootDistn = "normal",
                     wavFam = "DaubLeAsymm", wavFil = 8, wavBC = "periodic",
                     plot = TRUE, ...)
```

Arguments

`X`	vector of equally-spaced data. This must be of length $T = 2^J$ where $J > 2$ is an integer. It is required that `length(X)=length(Y)`. The vector should contain only numeric values and be non-missing. See the Details section for a description of the relationship between `X` and `Y`.

`Y`	vector of equally-spaced data. This must be of length $T = 2^J$ where $J > 2$ is an integer. It is required that `length(X)=length(Y)`. The vector should contain only numeric values and be non-missing. See the Details section for a description of the relationship between `X` and `Y`.
`J0`	wavelet coefficient threshold level. Allowed values are integers between 0 and $J - 2$. Note, J is related to the data series length: $T = 2^J$.
`R`	number of WiSE bootstrap samples. Allowed value is a positive integer. Default is 100.
`popParam`	hypothesized parameter values. Allowed input is a vector of length 2 which is non-missing and contains numeric entries. The first entry of the vector is the hypothesized value of the population intercept. The second entry of the vector is the hypothesized value of the population slope.
`XParam`	estimated linear parameter values (in time) from `X`. Allowed input is a vector of length 2 which is completely missing or contains numeric entries. The first entry of the vector is the intercept for `X` and the second entry is the slope for `X`. These are the estimated slope and intercept in the data – correspond to estimates of γ_{x0} and γ_{x1} in the Details section. If missing, these are estimated using least squares. If the data is modified with `padVector` or `padMatrix`, it is recommended that the user supply the `linearParam` output here.
`YParam`	estimated linear parameter values (in time) from `Y`. Allowed input is a vector of length 2 which is completely missing or contains numeric entries. The first entry of the vector is the intercept for `Y` and the second entry is the slope for `Y`. These are the estimated slope and intercept in the data – correspond to estimates of γ_{y0} and γ_{y1} in the Details section. If missing, these are estimated using least squares. If the data is modified with `padVector` or `padMatrix`, it is recommended that the user supply the `linearParam` output here.
`TauSq`	scale parameter for the bootstrap. Allowed values are `"log"`, `"log10"`, `"sqrt"`, `"1"`, or `"2/5"`. The scale parameter is related to the length of the data series. For example, `"log"` implies a value of the scale parameter, τ, of $\sqrt{log(T)}$. The value of `"1"` corresponds to the case of wild bootstrap.
`bootDistn`	the distribution for the bootstrap. Allowed values are `"normal"`, `"uniform"`, `"laplace"`, `"lognormal"`, `"gumbel"`, `"exponential"`, `"t5"`, `"t8"`, and `"t14"`. This draws iid random samples from the specified distribution for the wild bootstrap where the random variables have mean 0 and variance 1. For example, `"t5"` is Student's t-distribution with 5 degrees of freedom.
`wavFam`	wavelet family. Allowed values are `"DaubLeAsymm"` and `"DaubExPhase"` – Daubechies Least Asymmetric and Daubechies Extremal Phase. This is the `family` used within the `wavethresh` package.
`wavFil`	wavelet filter number. Allowed values are integers between 4 and 10 when `wavFam="DaubLeAsymm"` or integers between 1 and 10 when `wavFam="DaubExPhase"`. These correspond to the number of vanishing moments of the wavelet. This is the `filter.number` used within the `wavethresh` package.
`wavBC`	wavelet boundary condition. Allowed values are `"periodic"` and `"symmetric"`. This is the `bc` used within the `wavethresh` package.

`plot`	logical. If `TRUE`, a plot of the bootstrap sample of the linear parameters (generated under the null hypothesis) and the estimated parameters from the data is shown.
`...`	additional graphical arguments. See `plot`, `plot.default`.

Details

Given 2 vectors of equally-spaced data of length $T = 2^J$ for a positive integer, J, we assume the following models:

$$X = \gamma_{x0}1 + \gamma_{x1}t + W\gamma_x + e_x$$

$$Y = \gamma_{y0}1 + \gamma_{y1}t + W\gamma_y + e_y$$

where Y and X are the data vectors, linear parameters in time (t) are $\gamma_{x0}, \gamma_{x1}, \gamma_{y0}$ and γ_{y1}.

The γ_x and γ_y are the wavelet coefficients (scaling and filter) and W is the DWT for a fixed wavelet basis. Note, in many cases of the DWT, the scaling coefficient is equivalent to γ_{x0}, γ_{y0}, and thus, estimated there.

In this function, we consider a linear relationship between the wavelet coefficients. Specifically, we hypothesize a relationship

$$\gamma_y = \alpha 1 + \beta\gamma_x$$

The null hypothesis is

$$H_0 : \alpha = m, \beta = n$$

for real numbers m, n. The user specifies `popParam=c(m, n)`.

The WiSE bootstrap sample is created under the null hypothesis for a set threshold, `J0=j`. The sampling scheme is described in detail in Braverman et al. The distributon of the bootstrap sample of the parameters allows for calculation of a p-value associated with the null hypothesis.

Some notation to aid in understanding outputs:

1) a, b: estimates of α, β from the data wavelet coefficients

2) $a*, b*$: estimates of α, β from the bootstrap wavelet coefficients

3) g_{xj}, g_{yj}: estimates of γ_x, γ_y from the data at the threshold `J0=j`

4) $g*_{xj}, g*_{yj}$: estimates of γ_x, γ_y from the bootstrap sample at the threshold `J0=j`

Value

`AsymptoticPValue`	the asymptotic p-value based upon Hotelling's T^2.
`BootstrapPValue`	the bootstrap p-value.
`dataSlope`	the estimated slope of the wavelet coefficients from the data. In the notation from Details, b
`dataIntercept`	the estimated intercept of the wavelet coefficients from the data. In the notation from Details, a
`bootSlope`	the estimated slopes of the wavelet coefficients from the bootstrap samples. In the notation from Details, $b*$. This is a vector of length `R`.

`bootIntercept`	the estimated intercepts of the wavelet coefficients from the bootstrap samples. In the notation from Details, $a*$. This is a vector of length `R`.
`YWavelet`	the estimated wavelet coefficients from the `Y` data. In the notation from Details, g_{yj}. This is a vector of length $2^(J0+1)-1$. The first entry is the level 0 coefficient, ..., final entries are the level $J0$ coefficients.
`XWavelet`	the estimated wavelet coefficients from the `X` data. In the notation from Details, g_{xj}. This is a vector of length $2^(J0+1)-1$. The first entry is the level 0 coefficient, ..., final entries are the level $J0$ coefficients.
`bootYWavelet`	the estimated wavelet coefficients from the `Y` bootstrap sample. In the notation from Details, $g*_{yj}$. This is a matrix with `R` rows and $2^{J0+1}-1$ columns which correspond to the wavelet coefficients. The first column is the level 0 filter coefficient, ..., final columns are the level $J0$ filter coefficients.
`bootXWavelet`	the estimated wavelet coefficients from the `X` bootstrap sample. In the notation from Details, $g*_{xj}$. This is a matrix with `R` rows and $2^{J0+1}-1$ columns which correspond to the wavelet coefficients. The first column is the level 0 filter coefficient, ..., final columns are the level $J0$ filter coefficients.

Author(s)

Megan Heyman

References

The WiSE bootstrap hypothesis test is implemented as an analysis tool in Braverman, A. et al. "Probabilistic Climate Model Evaluation" (in progress).

See Also

`padMatrix`, `padVector`, `wavethresh-package`

Examples

```
##Test whether \alpha=0 and \beta=1 for AIRS and IPSL Run 1 at 60E
## R=10 bootstrap samples is not recommended.  For demonstration only.
data(CM20N20S60E)
padData <- padMatrix(CM20N20S60E)
hypTest <- WiSEHypothesisTest(padData$xPad[,1], padData$xPad[,2], J0=5, R=10,
                          XParam=padData$linearParam[,1], YParam=padData$linearParam[,2],
                               plot=TRUE)
```

Index