Package 'rwt'

June 14, 2022

Version 1.0.2

Date 2022-06-09

Title 'Rice Wavelet Toolbox' Wrapper

Description Provides a set of functions for 1D and 2D wavelet and filter bank design, analysis, and processing. Functions for denoising are also included. The package can be used for image denoising or deconvolution.

Depends R (>= 2.15)

Imports matlab

URL https://cran.r-project.org/package=rwt

License BSD_3_clause + file LICENSE

Copyright file COPYRIGHTS

NeedsCompilation yes

LazyLoad yes

Repository CRAN

Date/Publication 2022-06-14 11:40:02 UTC

Repository/R-Forge/Project estimate

Repository/R-Forge/Revision 61

Repository/R-Forge/DateTimeStamp 2022-06-09 14:42:11

Author P. Roebuck [aut, trl, cre] (R port),
Rice University's DSP group [aut, cph] (MATLAB toolbox),
MD Anderson Cancer Center [cph]

Maintainer P. Roebuck <proebuck1701@gmail.com>

R topics documented:

`rwt-package` *'Rice Wavelet Toolbox' Wrapper*

Description

Provides a set of functions for 1D and 2D wavelet and filter bank design, analysis, and processing. Functions for denoising are also included. The package can be used for image denoising or deconvolution.

Details

Package:	rwt
Type:	package
Version:	1.0.2
Date:	2022-06-09
License:	BSD_3_clause + file LICENSE

For a complete list of functions, use `library(help="rwt")`.
For a high-level summary of the changes for each revision, use `file.show(system.file("NEWS", package="rwt"))`.

Author(s)

P. Roebuck <proebuck1701@gmail.com>

`daubcqf` *Daubechies Filter Creation*

Description

Computes the Daubechies' scaling and wavelet filters (normalized to $sqrt(2)$).

Usage

```
daubcqf(N, type = PHASE.MINIMUM)
```

Arguments

N	numeric scalar specifying length of filter (must be even)
type	Distinguishes the minimum phase, maximum phase and mid-phase solutions. Valid values are:

```
PHASE.MINIMUM
PHASE.MID
PHASE.MAXIMUM
```

Value

Returns a list with components:

h.0	minimal phase Daubechies' scaling filter
h.1	minimal phase Daubechies' wavelet filter

Author(s)

P. Roebuck <proebuck1701@gmail.com>

References

Ingrid Daubechies
Orthonormal Bases of Compactly Supported Wavelets
CPAM (Oct. 1988), 7(41), 909-996
doi:10.1002/cpa.3160410705

Examples

```
h <- daubcqf(6)
```

---

denoise *Wavelet-based Denoising*

Description

Denoise the signal x using the 2-band wavelet system described by the filter h using either the traditional discrete wavelet transform (DWT) or the linear shift invariant discrete wavelet transform (also known as the undecimated DWT (UDWT)).

Usage

```
denoise(x, h, type, option)
denoise.dwt(x, h, option = default.dwt.option)
denoise.udwt(x, h, option = default.udwt.option)
```

Arguments

x	1D or 2D signal to be denoised
h	numeric scalar specifying scaling filter to be applied
type	type of transform. Valid values are: DWT.TRANSFORM.TYPE UDWT.TRANSFORM.TYPE
option	list containing desired transformation settings

Details

The transformation settings in the option list are:

threshold.low.pass.part: logical scalar. If TRUE, threshold the low-pass component.

threshold.multiplier: thld = c*MAD(noise_estimate)

variance.estimator: Valid values are:

MAD.VARIANCE.ESTIMATOR	Mean absolute deviation
STD.VARIANCE.ESTIMATOR	Classical numerical std estimate

threshold.type: Valid values are:

SOFT.THRESHOLD.TYPE	Soft thresholding
HARD.THRESHOLD.TYPE	Hard thresholding

num.decompression.levels: number of levels in wavelet decomposition. Setting this to MAX.DECOMPOSITION will allow maximal decomposition.

threshold: actual threshold to use. Setting this to anything but CALC.THRESHOLD.TO.USE will disable the variance.estimator setting.

Value

Returns a list with components:

xd	estimate of noise free signal
xn	estimated noise signal (x-xd)
option	list of actual parameters used. It is configured the same way as the input option list with an additional element - option[[7]] = type.

Note

Both denoise.dwt and denoise.udwt are convenience routines that call the denoise routine with appropriate default arguments.

Author(s)

P. Roebuck <proebuck1701@gmail.com>

Examples

```
sig <- makesig(SIGNAL.DOPPLER)
h <- daubcqf(6)
ret.dwt <- denoise.dwt(sig$x, h$h.0)
```

`makesig` *Make Signal*

Description

Creates artificial test signal identical to the standard test signals proposed and used by D. Donoho and I. Johnstone in WaveLab, a MATLAB toolbox for wavelet analysis developed at Stanford University.

Usage

```
makesig(sigName, N)
```

Arguments

`sigName`	character string specifying name of desired signal. Valid values are:

SIGNAL.ALL
SIGNAL.HEAVI.SINE
SIGNAL.BUMPS
SIGNAL.BLOCKS
SIGNAL.DOPPLER
SIGNAL.RAMP
SIGNAL.CUSP
SIGNAL.SING
SIGNAL.HI.SINE
SIGNAL.LO.SINE
SIGNAL.LIN.CHIRP
SIGNAL.TWO.CHIRP
SIGNAL.QUAD.CHIRP
SIGNAL.MISH.MASH
SIGNAL.WERNER.SORROWS (Heisenberg)
SIGNAL.LEOPOLD (Kronecker)

`N`	numeric scalar specifying length in samples of desired signal (512 by default)

Value

Returns a list with components:

`x`	vector (or matrix) of test signals
`N`	length of signal returned

Note

Using the value `SIGNAL.ALL.SIG` for *sigName* returns a matrix containing the vectors of all the other signals.

Author(s)

P. Roebuck <proebuck1701@gmail.com>

References

J. Buckheit, S. Chen, D. Donoho, I. Johnstone, & J. Scargle
WaveLab (version 0.700; 1995) [Computer software].
https://statweb.stanford.edu/~wavelab/WaveLab701.html

Examples

```
ret.sig <- makesig(SIGNAL.DOPPLER, 32)
```

`mdwt` *Discrete Wavelet Transform*

Description

Computes the discrete wavelet transform `y` for input signal `x` using the scaling filter `h`.

Usage

```
mdwt(x, h, L)
```

Arguments

`x`	Finite 1D or 2D signal (implicitly periodized)
`h`	Scaling filter to be applied
`L`	Number of levels in wavelet decomposition. In the case of a 1D signal, length(x) must be divisible by 2^L; in the case of a 2D signal, the row and the column dimension must be divisible by 2^L. If no argument is specified, a full DWT is returned for maximal possible L.

Value

Returns a list with components:

`y`	Wavelet transform of the signal
`L`	Number of levels in wavelet decomposition

Author(s)

P. Roebuck <proebuck1701@gmail.com>

Examples

```
sig <- makesig(SIGNAL.LIN.CHIRP, 8)
h <- daubcqf(4)
L <- 2
ret.mdwt <- mdwt(sig$x, h$h.0, L)
```

`midwt` *Inverse Discrete Wavelet Transform*

Description

Computes the inverse discrete wavelet transform `x` for input signal `y` using the scaling filter `h`.

Usage

```
midwt(y, h, L)
```

Arguments

`y`	Finite 1D or 2D signal (implicitly periodized)
`h`	Scaling filter to be applied
`L`	Number of levels in wavelet decomposition. In the case of a 1D signal, length(x) must be divisible by 2^L; in the case of a 2D signal, the row and the column dimension must be divisible by 2^L. If no argument is specified, a full DWT is returned for maximal possible L.

Value

Returns a list with components:

`x`	Periodic reconstructed signal
`L`	Number of levels in wavelet decomposition

Author(s)

P. Roebuck <proebuck1701@gmail.com>

Examples

```
sig <- makesig(SIGNAL.LIN.CHIRP, 8)
h <- daubcqf(4)
L <- 1
ret.mdwt <- mdwt(sig$x, h$h.0, L)
ret.midwt <- midwt(ret.mdwt$y, h$h.0, ret.mdwt$L)
```

`mirdwt` *Inverse Redundant Discrete Wavelet Transform*

Description

Computes the inverse redundant discrete wavelet transform `x` for input signal `y` using the scaling filter `h`. (Redundant means here that the sub-sampling after each stage of the forward transform has been omitted.)

Usage

```
mirdwt(yl, yh, h, L)
```

Arguments

`yl`	Lowpass component
`yh`	Highpass components
`h`	Scaling filter to be applied
`L`	Number of levels in wavelet decomposition. In the case of a 1D signal, length(yl) must be divisible by 2^L; in the case of a 2D signal, the row and the column dimension must be divisible by 2^L. If no argument is specified, a full DWT is returned for maximal possible L.

Value

Returns a list with components:

`x`	Finite length 1D or 2D signal
`L`	Number of levels in wavelet decomposition

Author(s)

P. Roebuck <proebuck1701@gmail.com>

Examples

```
sig <- makesig(SIGNAL.LEOPOLD, 8)
h <- daubcqf(4)
L <- 1
ret.mrdwt <- mrdwt(sig$x, h$h.0, L)
ret.mirdwt <- mirdwt(ret.mrdwt$yl, ret.mrdwt$yh, h$h.0, ret.mrdwt$L)
```

mrdwt *Redundant Discrete Wavelet Transform*

Description

Computes the redundant discrete wavelet transform y for input signal x using the scaling filter h. Redundant means here that the sub-sampling after each stage is omitted.

Usage

```
mrdwt(x, h, L)
```

Arguments

x	Finite 1D or 2D signal (implicitly periodized)
h	Scaling filter to be applied
L	Number of levels in wavelet decomposition. In the case of a 1D signal, length(x) must be divisible by 2^L; in the case of a 2D signal, the row and the column dimension must be divisible by 2^L. If no argument is specified, a full DWT is returned for maximal possible L.

Value

Returns a list with components:

yl	Lowpass component
yh	Highpass components
L	Number of levels in wavelet decomposition

Author(s)

P. Roebuck <proebuck1701@gmail.com>

Examples

```
sig <- makesig(SIGNAL.LEOPOLD, 8)
h <- daubcqf(4)
L <- 1
ret.mrdwt <- mrdwt(sig$x, h$h.0, L)
```

plotSignalTransformation *Plot Signal and its Transform*

Description

Plots the signal `s` and its transform `x` on graphics device.

Usage

```
plotSignalTransformation(x, s, title, col.x = 'blue', col.s = 'red')
```

Arguments

`x`	wavelet transformed signal to be plotted
`s`	original signal to be plotted
`title`	overall title for the plot
`col.x`	color to be used for plotting `x` values as lines
`col.s`	color to be used for plotting `s` values as lines

Details

Used by demo code to display the results of a transformation.

Value

Returns no value – called for its side effect.

Author(s)

P. Roebuck <proebuck1701@gmail.com>

threshold *Threshold Input Signal*

Description

Thresholds the input signal `y` with the threshold value `thld`.

Usage

```
hardTh(y, thld)
softTh(y, thld)
```

Arguments

y	1D or 2D signal to be thresholded
thld	numeric threshold value to be applied

Value

Returns numeric vector of thresholded values of the same length as y.

Author(s)

P. Roebuck <proebuck1701@gmail.com>

References

D.L. Donoho
De-noising via Soft-Thresholding
Tech. Rept. Statistics, Stanford (1992)

Examples

```
sig <- makesig(SIGNAL.WERNER.SORROWS, 8)
thld <- 1
x <- hardTh(sig$x, thld)
```

Index