Package ‘waved’

October 12, 2022

Version 1.2
License GPL


Title Wavelet Deconvolution

Author Marc Raimondo <marcr@maths.usyd.edu.au> and Michael Stewart <michael.stewart@sydney.edu.au>

Maintainer Michael Stewart <michael.stewart@sydney.edu.au>

URL http://www.jstatsoft.org/v21/i02

NeedsCompilation no

Repository CRAN

Date/Publication 2018-05-29 07:36:12 UTC

R topics documented:

FWaveD .......................................................... 1
WaveD .......................................................... 3
waved.example ............................................... 5

Index 7

<table>
<thead>
<tr>
<th>FWaveD</th>
<th>FWaveD</th>
</tr>
</thead>
</table>

Description

Computes the Forward WaveD Transform.
Usage

FWaveD(y,g=1,L=3,deg=3,F=(log2(length(y))-1),thr=rep(0,log2(length(y))),SOFT=FALSE)

Arguments

y
Sample of \( f \ast g \) + (Gaussian noise), a vector of dyadic length (i.e. \( 2^J-1 \) where \( J \)

is the largest resolution level). Here \( f \) is the target function, \( g \) is the convolution

kernel.

g
Sample of \( g \) or \( g \) + (Gaussian noise), same length as \( y \). The default is the

Dirac mass at 0.

L
Lowest resolution level; the default is 3.

deg
The degree of the Meyer wavelet, either 1, 2, or 3 (the default).

F
Finest resolution level; the default is the data-driven choice \( j_1 \) (see Value below).

thr
A vector of length \( F - L + 1 \), giving thresholds at each resolution levels \( L, L+1, \ldots, F \); default is maxiset threshold.

SOFT
if SOFT=TRUE, uses the soft thresholding policy as opposed to the hard (SOFT=FALSE, the default).

Value

Returns a vector of wavelet coefficients of length \( n \) (the same length as \( y \)), the last \( n/2 \) entries are

wavelet coefficients at resolution level \( J-1 \), where \( J = \log_2(n) \); the \( n/4 \) entries before that are

the wavelet coefficients at resolution level \( J-2 \), and so on until level \( L \). In addition the \( 2^L \) entries

are scaling coefficients at coarse level \( C = L \).

References


periodic setting’, *Journal of the Royal Statistical Society, Series B* 66(3),547–573. with discussion

pp.627–652.


matics and Statistics, University of Sydney.

See Also

WaveD

Examples

library(waved)
data=waved.example(TRUE,FALSE)

lidar.w=FWaveD(data$lidar.blur,data$g)
WaveD

Description

Performs statistical wavelet deconvolution using Meyer wavelet.

Usage

WaveD(yobs, g=c(1, rep(0, (length(yobs)-1))), MC=FALSE, SOFT=FALSE, 
F=find.j1(g, scale(yobs))[2], L=3, deg=3, eta=sqrt(6), 
thr=maxithresh(yobs, g, eta=eta), label="WaveD")

Arguments

- **yobs**: Sample of \( f \ast g \) + (Gaussian noise), a vector of dyadic length (i.e. \( 2^J - 1 \) where \( J \) is the largest resolution level). Here \( f \) is the target function, \( g \) is the convolution kernel.
- **g**: Sample of \( g \) or \( g + \) (Gaussian noise), same length as yobs. The default is the Dirac mass at 0.
- **MC**: Option to only return the (fast) translation-invariant WaveD estimate (MC=TRUE) as opposed to the full WaveD output (MC=FALSE, the default), as described below. MC=TRUE recommended for Monte Carlo simulation.
- **SOFT**: if SOFT=TRUE, uses the soft thresholding policy as opposed to the hard (SOFT=FALSE, the default).
- **F**: Finest resolution level; the default is the data-driven choice \( j_1 \) (see Value below).
- **L**: Lowest resolution level; the default is 3.
- **deg**: The degree of the Meyer wavelet, either 1, 2, or 3 (the default).
- **eta**: Tuning parameter of the maxiset threshold; default is \( \sqrt{6} \).
- **thr**: A vector of length \( F - L + 1 \), giving thresholds at each resolution levels \( L, L + 1, \ldots, F \); default is maxiset threshold.
- **label**: Auxiliary plotting parameter; do not change this.

Value

In the case that MC=TRUE, WaveD returns a vector consisting of the translation-invariant WaveD estimate. In the case that MC=FALSE (the default), WaveD returns a list with components

- **waved**: translation invariant WaveD transform; in the case MC=TRUE this is all that is returned.
- **ordinary**: ordinary WaveD transform
- **FWaveD**: Forward WaveD Transform; see FWaveD.
- **w**: alternate name for FWaveD
- **w.thr**: thresholded version of w
IWaveD Inverse WaveD Transform
iw alternate name for IWaveD
s estimate of the noise standard deviation
j1 estimate of optimal resolution level (for maxiset threshold).
F Fine resolution level used (may be different to j1).
M estimate of optimal Fourier frequency (for maxiset threshold).
thr vector of thresholds used (default is maxiset threshold).
percent percentage of thresholding per resolution level
noise noise proxy, wavelet coefficients of the raw data at the largest resolution level, used for estimating noise features.
ps P-value of the Shapiro-Wilk test for normality applied to the noise proxy.
residuals wavelet coefficients that have been removed before fine level F.

Author(s)
Marc Raimondo and Michael Stewart

References

See Also
FWaveD

Examples
library(waved)
data=waved.example(TRUE,FALSE)
doppler.wvd=WaveD(data$doppler.noisy,data$g)
summary(doppler.wvd)
Description

Generate data sets and figures to illustrate the WaveD function.

Usage

waved.example(pr = TRUE, gr=TRUE)

Arguments

pr If pr=TRUE (default) uses the same parameters as in the reference paper below. If pr=FALSE user level parameter specifications.
gr If gr=TRUE (default) text and graphical displays are provided.

Value

lidar.noisy Noisy blurred LIDAR signal (Gaussian noise)
lidar.noisyT Noisy blurred LIDAR signal (Student $t_2$ noise)
doppler.noisy Noisy blurred Doppler signal (Gaussian noise)
doppler.noisyT Noisy blurred Doppler signal (Student $t_2$ noise)
lidar.blur Blurred LIDAR signal
doppler.blur Blurred Doppler signal
t Rime vector scaled to [0,1]
n Sample size
g Convolution kernel
lidar LIDAR signal
doppler Doppler signal.
seed Used in set.seed
sigma Noise standard deviation.
g.noisy Convolution kernel plus Gaussian noise.
g.noisyT Convolution kernel plus Student $t_2$ noise.
dip Degree of Ill-posedness.
k.scale Scale of the convolution kernel

Author(s)

Marc Raimondo
References


See Also

WaveD

Examples

data = waved.example(TRUE, FALSE)
Index

* nonparametric
  FWaveD, 1
  WaveD, 3
  waved.example, 5

FWaveD, 1, 3, 4
WaveD, 2, 3, 6
waved.example, 5