Package ‘biwavelet’

September 1, 2016

Type Package
Title Conduct Univariate and Bivariate Wavelet Analyses
Version 0.20.11
Date 2016-08-31
Author Tarik C. Gouhier, Aslak Grinsted, Viliam Simko
Maintainer Tarik C. Gouhier <tarik.gouhier@gmail.com>
Description This is a port of the WTC MATLAB package written by Aslak Grinsted and the wavelet program written by Christopher Torrence and Gibert P. Compo. This package can be used to perform univariate and bivariate (cross-wavelet, wavelet coherence, wavelet clustering) analyses.
License GPL (>= 2)
URL https://github.com/tgouhier/biwavelet
BugReports https://github.com/tgouhier/biwavelet/issues
LazyData yes
LinkingTo Rcpp
Imports fields, foreach, Rcpp (>= 0.12.2)
Suggests testthat, knitr, rmarkdown, devtools, profvis, microbenchmark
VignetteBuilder knitr
RoxygenNote 5.0.1
NeedsCompilation yes
Repository CRAN
Date/Publication 2016-09-01 01:52:02

R topics documented:

  biwavelet-package ................................................. 2
  ar1.spectrum .................................................. 4
  ar1_ma0_sim .................................................. 5
  arrow .......................................................... 6
biwavelet-package

Conduct Univariate and Bivariate Wavelet Analyses

Description

This is a port of the WTC MATLAB package written by Aslak Grinsted and the wavelet program written by Christopher Torrence and Gibert P. Compo. This package can be used to perform univariate and bivariate (cross-wavelet, wavelet coherence, wavelet clustering) wavelet analyses.

Details

As of biwavelet version 0.14, the bias-corrected wavelet and cross-wavelet spectra are automatically computed and plotted by default using the methods described by Liu et al. (2007) and Veleda et al. (2012). This correction is needed because the traditional approach for computing the power spectrum (e.g., Torrence and Compo 1998) leads to an artificial and systematic reduction in power at lower periods.
Author(s)
Tarik C. Gouhier
Maintainer: Tarik C. Gouhier <tarik.gouhier@gmail.com>
Code based on WTC MATLAB package written by Aslak Grinsted and the wavelet MATLAB program written by Christopher Torrence and Gibert P. Compo.

References

Examples
# As of biwavelet version 0.14, the bias-corrected wavelet and cross-wavelet spectra
# are automatically computed and plotted by default using the methods
# described by Liu et al. (2007) and Veleda et al. (2012). This correction
# is needed because the traditional approach for computing the power spectrum
# (e.g., Torrence and Compo 1998) leads to an artificial and systematic reduction
# in power at low periods.

# EXAMPLE OF BIAS CORRECTION:
require(biwavelet)
# Generate a synthetic time series 's' with the same power at three distinct periods
t1=sin(seq(from=0, to=2*5*pi, length=1000))
t2=sin(seq(from=0, to=2*15*pi, length=1000))
t3=sin(seq(from=0, to=2*40*pi, length=1000))
s=t1+t2+t3

# Compare non-corrected vs. corrected wavelet spectrum
wt1=wt(cbind(t1:1000, s))
par(mfrow=c(1,2))
plot(wt1, type="power.corr.norm", main="Bias-corrected")
plot(wt1, type="power.norm", main="Not-corrected")

# ADDITIONAL EXAMPLES
t1 <- cbind(1:100, rnorm(100))
t2 <- cbind(1:100, rnorm(100))

# Continuous wavelet transform
wt.t1 <- wt(t1)

# Plot power
# Make room to the right for the color bar
par(oma = c(0, 0, 0, 1), mar = c(5, 4, 4, 5) + 0.1)
plot(wt.t1, plot.cb=TRUE, plot.phase=FALSE)

# Compute cross-wavelet
xwt.t1t2 <- xwt(t1, t2)

# Plot cross wavelet power and phase difference (arrows)
plot(xwt.t1t2, plot.cb=TRUE)

# Wavelet coherence; nrands should be large (> 1000)
wtc.t1t2 = wtc(t1, t2, nrands=10)
# Plot wavelet coherence and phase difference (arrows)
# Make room to the right for the color bar
par(oma=c(0, 0, 0, 1), mar=c(5, 4, 4, 5) + 0.1)
plot(wtc.t1t2, plot.cb=TRUE)

# Perform wavelet clustering of three time series
t1=cbind(1:100, sin(seq(from=0, to=10*2*pi, length.out=100)))
t2=cbind(1:100, sin(seq(from=0, to=10*2*pi, length.out=100)+0.1*pi))
t3=cbind(1:100, rnorm(100))

# Compute wavelet spectra
wt.t1=wt(t1)
wt.t2=wt(t2)
wt.t3=wt(t3)

# Store all wavelet spectra into array
w.arr=array(NA, dim=c(3, NROW(wt.t1$wave), NCOL(wt.t1$wave)))
w.arr[1, ] = wt.t1$wave
w.arr[2, ] = wt.t2$wave
w.arr[3, ] = wt.t3$wave

# Compute dissimilarity and distance matrices
w.arr.dis <- wclust(w.arr)
plot(hclust(w.arr.dis$dist.mat, method = "ward.D"), sub = "", main = "",
ylab = "Dissimilarity", hang = -1)
**ar1_ma0_sim**

**Description**
Generate the power spectrum of a random time series with a specific AR(1) coefficient

**Usage**
ar1.spectrum(ar1, periods)

**Arguments**
ar1       first order coefficient desired.
periods   periods of the time series at which the spectrum should be computed.

**Value**
Returns the power spectrum.

**Author(s)**
Tarik C. Gouhier (tarik.gouhier@gmail.com) Code based on WTC MATLAB package written by Aslak Grinsted.

**References**

**Examples**
p <- ar1.spectrum(0.5, 1:25)

---

**ar1_ma0_sim**

Slightly faster arima.sim implementation which assumes AR(1) and ma=0.

**Description**
Slightly faster arima.sim implementation which assumes AR(1) and ma=0.

**Usage**
ar1_ma0_sim(minroots, ar, n)
Arguments

- **minroots**: Output from `get_minroots` function.
- **ar**: The 'ar' part of AR(1)
- **n**: Length of output series, before un-differencing. A strictly positive integer.

See Also

- `arima.sim`

---

**arrow**

**Helper function for phase.plot**

Description

Helper function for phase.plot

Usage

```
arrow(x, y, l = 0.1, w = 0.3 * l, alpha, col = "black")
```

Arguments

- **x**: TODO
- **y**: TODO
- **l**: TODO
- **w**: TODO
- **alpha**: TODO
- **col**: TODO

---

**check.data**

**Check the format of time series**

Description

Check the format of time series

Usage

```
check.data(y, x1 = NULL, x2 = NULL)
```
check.datum

Arguments

y time series y in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
x1 time series x1 in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
x2 time series x2 in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.

Value

Returns a named list containing:
t time steps
dt size of a time step
n.obs number of observations

Author(s)

Tarik C. Gouhier (tarik.gouhier@gmail.com)

References


Examples

t1 <- cbind(1:100, rnorm(100))
check.data(y = t1)

check.datum Helper function

Description

Helper function

Usage

check.datum(x)

Arguments

x matrix
convolve2D

Value

\[ \text{list}(t, \ dt, \ n\text{.obs}) \]

Note

This function is not exported

---

convolve2D

Fast column-wise convolution of a matrix

Description

Use the Fast Fourier Transform to perform convolutions between a sequence and each column of a matrix.

Usage

\[
\text{convolve2D}(x, \ y, \ \text{conj} = \text{TRUE}, \ \text{type} = \text{c("circular", "open")})
\]

Arguments

- **x**: M \times n matrix.
- **y**: Numeric sequence of length N.
- **conj**: Logical; if TRUE, take the complex conjugate before back-transforming. Default is TRUE and used for usual convolution.
- **type**: Character; one of circular, open (beginning of word is ok).

For circular, the two sequences are treated as circular, i.e., periodic.

For open and filter, the sequences are padded with zeros (from left and right) first; filter returns the middle sub-vector of open, namely, the result of running a weighted mean of x with weights y.

Details

This is a corrupted version of convolve made by replacing \text{fft} with \text{mvfft} in a few places. It would be nice to submit this to the R Developers for inclusion.

Value

M \times n matrix

Note

This function was copied from \text{waveslim} to limit package dependencies.

Author(s)

Brandon Whitcher
**convolve2D_typeopen**

*Speed-optimized version of convolve2D*

**Description**

Equivalent to `convolve2D(x, y, type = "open")` The motivation for this function was that convolution is called many times in a loop from `smooth.wavelet`, always with the `type="open"` parameter.

**Usage**

`convolve2D_typeopen(x, y)`

**Arguments**

- `x` M x n matrix.
- `y` Numeric sequence of length N.

**Author(s)**

Viliam Simko

**See Also**

`convolve2D`

---

**enviro.data**

*Multivariate ENSO (MEI), NPGO, and PDO indices*

**Description**

Monthly indices of ENSO, NPGO, and PDO from 1950 to 2009

**Usage**

`data(enviro.data)`

**Format**

A data frame with 720 observations on the following 6 variables.

- `month` a numeric vector containing the month
- `year` a numeric vector containing the year
- `date` a numeric vector containing the date
- `mei` a numeric vector containing the MEI index
- `npgo` a numeric vector containing the NPGO index
- `pdo` a numeric vector containing the PDO index
get_minroots

Source

MEI: http://www.esrl.noaa.gov/psd/enso/mei
NPGO: http://www.o3d.org/npgo
PDO: http://jisao.washington.edu/pdo

References


Examples

data(enviro.data)
head(enviro.data)

get_minroots  Helper function

Description

Helper function

Usage

get_minroots(ar)

Arguments

ar       The ’ar’ part of AR(1)

Value

double
Supported mother wavelets

Description
The list of supported mother wavelets is used in multiple places therefore, we provide it as a lazily evaluated promise.

Usage

MOTHERS

Format
An object of class character of length 3.

---

phase.plot

Plot phases with arrows

Description
Plot phases with arrows

Usage

phase.plot(x, y, phases, arrow.len = min(par()$pin[2]/30, par()$pin[1]/40),
          arrow.col = "black", arrow.lwd = arrow.len * 0.3)

Arguments

x x-coordinates
y y-coordinates
phases phases
arrow.len size of the arrows. Default is based on plotting region (min(par()$pin[2]/30,par()$pin[1]/40).
arrow.col arrow line color. Default is black.
arrow.lwd width/thickness of arrows. Default is arrow.len * 0.3.

Note
Arrows pointing to the right mean that x and y are in phase.
Arrows pointing to the left mean that x and y are in anti-phase.
Arrows pointing up mean that y leads x by π/2.
Arrows pointing down mean that x leads y by π/2.
Author(s)

Tarik C. Gouhier (tarik.gouhier@gmail.com)
Huidong Tian provided a much better implementation of the phase.plot function that allows for more accurate phase arrows.
Original code based on WTC MATLAB package written by Aslak Grinsted.

Examples

# Not run: phase.plot(x, y, phases)

plot.biwavelet  

Plot biwavelet objects

Description

Plot biwavelet objects such as the cwt, cross-wavelet and wavelet coherence

Usage

## S3 method for class 'biwavelet'
plot(x, ncol = 64, fill.cols = NULL, xlab = "Time", ylab = "Period", tol = 1, plot.cb = FALSE, plot.phase = FALSE, type = "power.corr.norm", plot.coi = TRUE, lwd.coi = 1, col.coi = "white", lty.coi = 1, alpha.coi = 0.5, plot.sig = TRUE, lwd.sig = 4, col.sig = "black", lty.sig = 1, bw = FALSE, legend.loc = NULL, legend.horiz = FALSE, arrow.len = min(par()$pin[2]/30, par()$pin[1]/40), arrow.lwd = arrow.len * 0.3, arrow.cutoff = 1, arrow.col = "black", xlim = NULL, ylim = NULL, zlim = NULL, xaxt = "s", yaxt = "s", form = "%Y", ...)
plot.plot biwavelet

plot.cb plot color bar if TRUE. Default is FALSE.
plot.phase Plot phases with black arrows. Default is FALSE.
type type of plot to create. Can be power to plot the power, power.cor to plot the bias-corrected power, power.norm to plot the power normalized by the variance, power.cor.norm to plot the bias-corrected power normalized by the variance, wavelet to plot the wavelet coefficients, or phase to plot the phase. Default is power.cor.norm.
plot.coi plot cone of influence (COI) as a semi-transparent polygon if TRUE. Default is TRUE. Areas that fall within the polygon can be affected by edge effects.
lwd.coi Line width of COI. Default is 1.
col.coi Color of COI. Default is white.
lty.coi Line type of COI. Default is 1 for solide lines.
alpha.coi Transparency of COI. Range is 0 (full transparency) to 1 (no transparency). Default is 0.5.
plot.sig plot contours for significance if TRUE. Default is TRUE.
lwd.sig Line width of significance contours. Default is 4.
col.sig Color of significance contours. Default is black.
lty.sig Line type of significance contours. Default is 1.
bw plot in black and white if TRUE. Default is FALSE.
legend.loc legend location coordinates as defined by image.plot. Default is NULL.
legend.horiz plot a horizontal legend if TRUE. Default is FALSE.
arraow.len size of the arrows. Default is based on plotting region (min(par)$pin[2]/30,par$p[1]/40).
arraow.lwd width/thickness of arrows. Default is arrow.len*0.3.
arraow.cutoff cutoff value for plotting phase arrows. Phase arrows will be be plotted in regions where the significance of the zvalues exceeds arrow.cutoff. If the object being plotted does not have a significance field, regions whose zvalues exceed the arrow.cutoff quantile will be plotted. Default is 1.
arraow.col Color of arrows. Default is black.
xlim the x limits. The default is NULL.
ylim the y limits. The default is NULL.
zlim the z limits. The default is NULL.
xaxt Add x-axis? The default is s; use n for none.
yaxt Add y-axis? The default is s; use n for none.
form format to use to display dates on the x-axis. Default is '%Y' for 4-digit year. See ?date for other valid formats.
... other parameters.

Details

Arrows pointing to the right mean that x and y are in phase.
Arrows pointing to the left mean that x and y are in anti-phase.
Arrows pointing up mean that x leads y by π/2.
Arrows pointing down mean that y leads x by π/2.
Author(s)
Tarik C. Gouhier (tarik.gouhier@gmail.com) Code based on WTC MATLAB package written by Aslak Grinsted.

References

See Also
image.plot

Examples

```r
library(biwavelet)

t1 <- cbind(1:100, rnorm(100))
t2 <- cbind(1:100, rnorm(100))

# Continuous wavelet transform
wt.t1 <- wt(t1)

# Plot power
# Make room to the right for the color bar
par oma = c(0, 0, 0, 1), mar = c(5, 4, 4, 5) + 0.1
plot(wt.t1, plot.cb = TRUE, plot.phase = FALSE)

# Cross-wavelet transform
xwt.t1t2 <- xwt(t1, t2)

# Plot cross-wavelet
par oma = c(0, 0, 0, 1), mar = c(5, 4, 4, 5) + 0.1
plot(xwt.t1t2, plot.cb = TRUE)

# Example of bias-correction
t1 <- sin(seq(0, 2 * 5 * pi, length.out = 1000))
t2 <- sin(seq(0, 2 * 15 * pi, length.out = 1000))
t3 <- sin(seq(0, 2 * 40 * pi, length.out = 1000))

# This aggregate time series should have the same power
# at three distinct periods
s <- t1 + t2 + t3
```
Compute partial wavelet coherence

Description

Compute partial wavelet coherence between y and x1 by partialling out the effect of x2

Usage

```
pwtc(y, x1, x2, pad = TRUE, dj = 1/12, s0 = 2 * dt, J1 = NULL, max.scale = NULL, mother = "morlet", param = -1, lag1 = NULL, sig.level = 0.95, sig.test = 0, nrands = 300, quiet = FALSE)
```

Arguments

- **y**: time series y in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
- **x1**: time series x1 in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
- **x2**: time series x2 whose effects should be partialled out in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
- **pad**: pad the values will with zeros to increase the speed of the transform. Default is TRUE.
- **dj**: spacing between successive scales. Default is 1/12.
- **s0**: smallest scale of the wavelet. Default is 2*dt.
- **J1**: number of scales - 1.
- **max.scale**: maximum scale. Computed automatically if left unspecified.
mother: type of mother wavelet function to use. Can be set to morlet, dog, or paul. Default is morlet. Significance testing is only available for morlet wavelet.

param: nondimensional parameter specific to the wavelet function.

lag1: vector containing the AR(1) coefficient of each time series.

sig.level: significance level. Default is 0.95.

sig.test: type of significance test. If set to 0, use a regular $\chi^2$ test. If set to 1, then perform a time-average test. If set to 2, then do a scale-average test.

nrands: number of Monte Carlo randomizations. Default is 300.

quiet: Do not display progress bar. Default is FALSE.

Value

Return a biwavelet object containing:

- coi: matrix containing cone of influence
- wave: matrix containing the cross-wavelet transform of y and x1
- rsq: matrix of partial wavelet coherence between y and x1 (with x2 partialled out)
- phase: matrix of phases between y and x1
- period: vector of periods
- scale: vector of scales
- dt: length of a time step
- t: vector of times
- xaxis: vector of values used to plot xaxis
- s0: smallest scale of the wavelet
- dj: spacing between successive scales
- y.sigma: standard deviation of y
- x1.sigma: standard deviation of x1
- mother: mother wavelet used
- type: type of biwavelet object created (pwtc)
- signif: matrix containing sig.level percentiles of wavelet coherence based on the Monte Carlo AR(1) time series

Note

The Monte Carlo randomizations can be extremely slow for large datasets. For instance, 1000 randomizations of a dataset consisting of 1000 samples will take ~30 minutes on a 2.66 GHz dual-core Xeon processor.

Author(s)

Tarik C. Gouhier (tarik.gouhier@gmail.com) Code based on WTC MATLAB package written by Aslak Grinsted.
**rcpp_row_quantile**

**Row-wise quantile of a matrix**

**Description**

This is a C++ speed-optimized version. It is equivalent to R version `quantile(data, q, na.rm = TRUE)`

**References**


**Examples**

```r
library(biwavelet)

y <- cbind(1:100, rnorm(100))
x1 <- cbind(1:100, rnorm(100))
x2 <- cbind(1:100, rnorm(100))

# Partial wavelet coherence of y and x1
pwtc.yx1 <- pwtc(y, x1, x2, nrands = 0)

# Partial wavelet coherence of y and x2
pwtc.yx2 <- pwtc(y, x2, x1, nrands = 0)

# Plot partial wavelet coherence and phase difference (arrows)
# Make room to the right for the color bar
par(mfrow = c(2, 1), oma = c(4, 0, 0, 1),
     mar = c(1, 4, 4, 5), mgp = c(1.5, 0.5, 0))

plot(pwtc.yx1, xlab = "", plot.cb = TRUE,
     main = "Partial wavelet coherence of y and x1 | x2")

plot(pwtc.yx2, plot.cb = TRUE,
     main = "Partial wavelet coherence of y and x2 | x1")
```
Usage

rcpp_row_quantile(data, q)

Arguments

data Numeric matrix whose row quantiles are wanted.
q Probability with value in [0,1]

Value

A vector of length nrow(data), where each element represents row quantile.

Author(s)

Viliam Simko

rcpp_wt_bases_dog

Optimized "wt.bases.dog" function.

Description

This is a C++ version optimized for speed. Computes the wavelet as a function of Fourier frequency for "dog" mother wavelet.

Usage

rcpp_wt_bases_dog(k, scale, param = -1L)

Arguments

k vector of frequencies at which to calculate the wavelet.
scale the wavelet scale.
param nondimensional parameter specific to the wavelet function.

Value

Returns a list containing:
daughter wavelet function
fourier.factor ratio of fourier period to scale
coi cone of influence
dof degrees of freedom for each point in wavelet power

Note

This c++ implementation is approx. 50
Author(s)

Viliam Simko

rcpp_wt_bases_morlet

Optimized "wt.bases.morlet" function.

Description

This is a C++ version optimized for speed. Computes the wavelet as a function of Fourier frequency for "morlet" mother wavelet.

Usage

rcpp_wt_bases_morlet(k, scale, param = -1L)

Arguments

k vector of frequencies at which to calculate the wavelet.
scale the wavelet scale.
param nondimensional parameter specific to the wavelet function.

Value

Returns a list containing:

daughter wavelet function

decay.factor ratio of fourier period to scale

coi cone of influence

dof degrees of freedom for each point in wavelet power

Note

This c++ implementation is approx. 60

Author(s)

Viliam Simko
Description

This is a C++ version optimized for speed. Computes the wavelet as a function of Fourier frequency for "paul" mother wavelet.

Usage

rcpp_wt_bases_paul(k, scale, param = -1L)

Arguments

k vector of frequencies at which to calculate the wavelet.
scale the wavelet scale.
param nondimensional parameter specific to the wavelet function.

Value

Returns a list containing:

daughter wavelet function
fourier.factor ratio of fourier period to scale
coi cone of influence
dof degrees of freedom for each point in wavelet power

Note

This c++ implementation is approx. 59

Author(s)

Viliam Simko
smooth.wavelet

Smooth wavelet in both the time and scale domains

Description

The time smoothing uses a filter given by the absolute value of the wavelet function at each scale, normalized to have a total weight of unity, which is a Gaussian function for the Morlet wavelet. The scale smoothing is done with a boxcar function of width 0.6, which corresponds to the decorrelation scale of the Morlet wavelet.

Usage

smooth.wavelet(wave, dt, dj, scale)

Arguments

wave wavelet coefficients
dt size of time steps
dj number of octaves per scale
scale wavelet scales

Value

Returns the smoothed wavelet.

Note

This function is used internally for computing wavelet coherence. It is only appropriate for the morlet wavelet.

Author(s)

Tarik C. Gouhier (tarik.gouhier@gmail.com)

Code based on WTC MATLAB package written by Aslak Grinsted.

References


Examples

# Not run: smooth.wt1 <- smooth.wavelet(wave, dt, dj, scale)
wclust

*Compute dissimilarity between multiple wavelet spectra*

**Description**

Compute dissimilarity between multiple wavelet spectra

**Usage**

`wclust(w.arr, quiet = FALSE)`

**Arguments**

- `w.arr` N × p × t array of wavelet spectra where N is the number of wavelet spectra to be compared, p is the number of periods in each wavelet spectrum and t is the number of time steps in each wavelet spectrum.
- `quiet` Do not display progress bar. Default is FALSE

**Value**

Returns a list containing:

- `diss.mat` square dissimilarity matrix
- `dist.mat` (lower triangular) distance matrix

**Author(s)**

Tarik C. Gouhier (tarik.gouhier@gmail.com)

**References**


**Examples**

```r
library(biwavelet)
t1 <- cbind(1:100, sin(seq(0, 10 * 2 * pi, length.out = 100)))
t2 <- cbind(1:100, sin(seq(0, 10 * 2 * pi, length.out = 100) + 0.1 * pi))
t3 <- cbind(1:100, rnorm(100))  # white noise

# Compute wavelet spectra
wt.t1 <- wt(t1)
wt.t2 <- wt(t2)
```
wdist <- wt(t3)

## Store all wavelet spectra into array
w.arr <- array(dim = c(3, NROW(wt.t1$wave), NCOL(wt.t1$wave)))
w.arr[1, , ] <- wt.t1$wave
w.arr[2, , ] <- wt.t2$wave
w.arr[3, , ] <- wt.t3$wave

## Compute dissimilarity and distance matrices
w.arr.dis <- wclust(w.arr)
plot(hclust(w.arr.dis$dist.mat, method = "ward.D"),
     sub = "", main = "", ylab = "Dissimilarity", hang = -1)

wdist

*Compute dissimilarity between two wavelet spectra*

**Description**

Compute dissimilarity between two wavelet spectra

**Usage**

wdist(wt1, wt2, cutoff = 0.99)

**Arguments**

- **wt1**: power, wave or rsq matrix from biwavelet object generated by wt, xwt, or wtc.
- **wt2**: power, wave or rsq matrix from biwavelet object generated by wt, xwt, or wtc.
- **cutoff**: cutoff value used to compute dissimilarity. Only orthogonal axes that contribute more than 1 - cutoff to the total covariance between the two wavelet spectra will be used to compute their dissimilarity. Default is 0.99.

**Value**

Returns wavelet dissimilarity.

**Author(s)**

Tarik C. Gouhier (tarik.gouhier@gmail.com)

**References**


Examples

library(biwavelet)

t1 <- cbind(1:100, sin(seq(0, 10 * 2 * pi, length.out = 100)))
t2 <- cbind(1:100, sin(seq(0, 10 * 2 * pi, length.out = 100) + 0.1 * pi))

# Compute wavelet spectra
wt.t1 <- wt(t1)
wt.t2 <- wt(t2)

# Compute dissimilarity
wdist(wt.t1$wave, wt.t2$wave)

---

wt

*Compute wavelet transform*

Description

Compute wavelet transform

Usage

wt(d, pad = TRUE, dt = NULL, dj = 1/12, s0 = 2 * dt, J1 = NULL, max.scale = NULL, mother = "morlet", param = -1, lag1 = NULL, sig.level = 0.95, sig.test = 0, do.sig = TRUE)

Arguments

d
  time series in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.

pad
  pad the values will with zeros to increase the speed of the transform. Default is TRUE.

dt
  length of a time step.

dj
  spacing between successive scales. Default is 1/12.

s0
  smallest scale of the wavelet. Default is 2*dt

J1
  number of scales - 1.

max.scale
  maximum scale. Computed automatically if left unspecified.

mother
  type of mother wavelet function to use. Can be set to morlet, dog, or paul. Default is morlet.

param
  nondimensional parameter specific to the wavelet function.

lag1
  AR(1) coefficient of time series used to test for significant patterns.

sig.level
  significance level. Default is 0.95.

sig.test
  type of significance test. If set to 0, use a regular \( \chi^2 \) test. If set to 1, then perform a time-average test. If set to 2, then do a scale-average test.

do.sig
  perform significance testing if TRUE. Default is TRUE.
Value

Returns a biwavelet object containing:

coi  matrix containing cone of influence
wave matrix containing the wavelet transform
power matrix of power
power.corr matrix of bias-corrected power using the method described by Liu et al. (2007)
phase matrix of phases
period vector of periods
scale vector of scales
dt length of a time step
t vector of times
xaxis vector of values used to plot xaxis
s0 smallest scale of the wavelet
dj spacing between successive scales
sigma2 variance of time series
mother mother wavelet used
type type of biwavelet object created (wt)
signif matrix containing significance levels

Author(s)

Tarik C. Gouhier (tarik.gouhier@gmail.com)

Code based on wavelet MATLAB program written by Christopher Torrence and Gibert P. Compo.

References


Examples

t1 <- cbind(1:100, rnorm(100))

## Continuous wavelet transform
wt.t1 <- wt(t1)

## Plot power
## Make room to the right for the color bar
par(oma = c(0, 0, 0, 1), mar = c(5, 4, 4, 5) + 0.1)
plot(wt.t1, plot.cb = TRUE, plot.phase = FALSE)
wt.bases

**Compute wavelet**

**Description**
Computes the wavelet as a function of Fourier frequency.

**Usage**
```
wt.bases(mother = "morlet", ...)
```

**Arguments**
- **mother**: type of mother wavelet function to use. Can be set to morlet, dog, or paul. Default is morlet.
- **...**: see parameters k, scale and param in functions: `wt.bases.morlet`, `wt.bases.paul` and `wt.bases.dog`

**Value**
Returns a list containing:
- **daughter**: wavelet function
- **fourier.factor**: ratio of fourier period to scale
- **coi**: cone of influence
- **dof**: degrees of freedom for each point in wavelet power

**Author(s)**
Tarik C. Gouhier (tarik.gouhier@gmail.com)

Code based on wavelet MATLAB program written by Christopher Torrence and Gibert P. Compo.

**References**

**Examples**
```
# Not run: wb <- wt.bases(mother, k, scale[a1], param)
```
wt.bases.dog

*Helper method (not exported)*

**Description**

Helper method (not exported)

**Usage**

```r
wt.bases.dog(k, scale, param = -1)
```

**Arguments**

- `k`: vector of frequencies at which to calculate the wavelet.
- `scale`: the wavelet scale.
- `param`: nondimensional parameter specific to the wavelet function.

**Value**

Returns a list containing:

- `daughter`: wavelet function
- `fourier.factor`: ratio of fourier period to scale
- `coi`: cone of influence
- `dof`: degrees of freedom for each point in wavelet power

---

wt.bases.morlet

*Helper method (not exported)*

**Description**

Helper method (not exported)

**Usage**

```r
wt.bases.morlet(k, scale, param = -1)
```

**Arguments**

- `k`: vector of frequencies at which to calculate the wavelet.
- `scale`: the wavelet scale.
- `param`: nondimensional parameter specific to the wavelet function.
wt.bases.paul

Value

Returns a list containing:

- **daughter**: wavelet function
- **fourier.factor**: ratio of fourier period to scale
- **coi**: cone of influence
- **dof**: degrees of freedom for each point in wavelet power

Description

Helper method (not exported)

Usage

wt.bases.paul(k, scale, param = -1)

Arguments

- **k**: vector of frequencies at which to calculate the wavelet.
- **scale**: the wavelet scale.
- **param**: nondimensional parameter specific to the wavelet function.

Value

Returns a list containing:

- **daughter**: wavelet function
- **fourier.factor**: ratio of fourier period to scale
- **coi**: cone of influence
- **dof**: degrees of freedom for each point in wavelet power
wt.sig

Determine significance of wavelet transform

Description

Determine significance of wavelet transform

Usage

wt.sig(d, dt, scale, sig.test = 0, sig.level = 0.95, dof = 2, 
        lag1 = NULL, mother = "morlet", param = -1, sigma2 = NULL)

Arguments

d            time series in matrix format (n rows x 2 columns). The first column should 
              contain the time steps and the second column should contain the values.
dt           length of a time step.
scale        the wavelet scale.
sig.test     type of significance test. If set to 0, use a regular $\chi^2$ test. If set to 1, then perform 
              a time-average test. If set to 2, then do a scale-average test.
sig.level    significance level. Default is 0.95.
dof          degrees of freedom for each point in wavelet power.
lag1         AR(1) coefficient of time series used to test for significant patterns.
mother       type of mother wavelet function to use. Can be set to morlet, dog, or paul. 
              Default is morlet.
param        nondimensional parameter specific to the wavelet function.
sigma2       variance of time series

Value

Returns a list containing:

signif       vector containing significance level for each scale
signif       vector of red-noise spectrum for each period

Author(s)

Tarik C. Gouhier (tarik.gouhier@gmail.com)

Code based on wavelet MATLAB program written by Christopher Torrence and Gibert P. Compo.

References

American Meteorological Society 79:61-78.
Examples

# Not run: wt.sig(d, dt, scale, sig.test, sig.level, lag1,
#      dof = -1, mother = "morlet", sigma2 = 1)

wtc

Compute wavelet coherence

Description

Compute wavelet coherence

Usage

wtc(d1, d2, pad = TRUE, dj = 1/12, s0 = 2 * dt, J1 = NULL,
max.scale = NULL, mother = "morlet", param = -1, lag1 = NULL,
sig.level = 0.95, sig.test = 0, nrands = 300, quiet = FALSE)

Arguments

d1

d2

pad

dj

s0

J1

max.scale

mother

param

lag1

sig.level

sig.test

nrands

quiet
time series 1 in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
time series 2 in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
the values will with zeros to increase the speed of the transform. Default is TRUE.
spacing between successive scales. Default is 1/12.
smallest scale of the wavelet. Default is 2*dt.
number of scales - 1.
maximum scale. Computed automatically if left unspecified.
type of mother wavelet function to use. Can be set to morlet, dog, or paul. Default is morlet. Significance testing is only available for morlet wavelet.
nondimensional parameter specific to the wavelet function.
vector containing the AR(1) coefficient of each time series.
significance level. Default is 0.95.
type of significance test. If set to 0, use a regular $\chi^2$ test. If set to 1, then perform a time-average test. If set to 2, then do a scale-average test.
number of Monte Carlo randomizations. Default is 300.
Do not display progress bar. Default is FALSE
**Value**

Return a biwavelet object containing:

- `coi`: matrix containing cone of influence
- `wave`: matrix containing the cross-wavelet transform
- `wave.corr`: matrix containing the bias-corrected cross-wavelet transform using the method described by Veleda et al. (2012)
- `power`: matrix of power
- `power.corr`: matrix of bias-corrected cross-wavelet power using the method described by Veleda et al. (2012)
- `rsq`: matrix of wavelet coherence
- `phase`: matrix of phases
- `period`: vector of periods
- `scale`: vector of scales
- `dt`: length of a time step
- `t`: vector of times
- `xaxis`: vector of values used to plot xaxis
- `s0`: smallest scale of the wavelet
- `dj`: spacing between successive scales
- `d1.sigma`: standard deviation of time series 1
- `d2.sigma`: standard deviation of time series 2
- `mother`: mother wavelet used
- `type`: type of biwavelet object created (wtc)
- `signif`: matrix containing sig.level percentiles of wavelet coherence based on the Monte Carlo AR(1) time series

**Note**

The Monte Carlo randomizations can be extremely slow for large datasets. For instance, 1000 randomizations of a dataset consisting of 1000 samples will take ~30 minutes on a 2.66 GHz dual-core Xeon processor.

**Author(s)**

Tarik C. Gouhier (tarik.gouhier@gmail.com)

Code based on WTC MATLAB package written by Aslak Grinsted.
References


Examples

```r
# t1 <- cbind(1:100, rnorm(100))
# t2 <- cbind(1:100, rnorm(100))

## Wavelet coherence
wtc.tlt2 <- wtc(t1, t2, nrands = 10)

## Plot wavelet coherence and phase difference (arrows)
## Make room to the right for the color bar
par(oma = c(0, 0, 0, 1), mar = c(5, 4, 4, 5) + 0.1)
plot(wtc.tlt2, plot.cb = TRUE, plot.phase = TRUE)
```

---

**wtc.sig**  
*Determine significance of wavelet coherence*

Description

Determine significance of wavelet coherence

Usage

```
wtc.sig(nrands = 300, lag1, dt, ntimesteps, pad = TRUE, dj = 1/12, s0, J1,
         max.scale = NULL, mother = "morlet", sig.level = 0.95, quiet = FALSE)
```

Arguments

- **nrands**: number of Monte Carlo randomizations. Default is 300.
- **lag1**: vector containing the AR(1) coefficient of each time series.
- **dt**: length of a time step.
- **ntimesteps**: number of time steps in time series.
- **pad**: pad the values will with zeros to increase the speed of the transform. Default is TRUE.
wtc.sig

**dj**

spacing between successive scales. Default is 1/12.

**s0**

smallest scale of the wavelet. Default is 2*dt

**J1**

number of scales - 1.

**max.scale**

maximum scale

**mother**

type of mother wavelet function to use. Can be set to morlet, dog, or paul. Default is morlet. Significance testing is only available for morlet wavelet.

**sig.level**

significance level to compute. Default is 0.95

**quiet**

Do not display progress bar. Default is FALSE

**Value**

Returns significance matrix containing the sig.level percentile of wavelet coherence at each time step and scale.

**Note**

The Monte Carlo randomizations can be extremely slow for large datasets. For instance, 1000 randomizations of a dataset consisting of 1000 samples will take ~30 minutes on a 2.66 GHz dual-core Xeon processor.

**Author(s)**

Tarik C. Gouhier (tarik.gouhier@gmail.com)

Code based on WTC MATLAB package written by Aslak Grinsted.

**References**


**Examples**

```r
# Not run: wtcSig <- wtc.sig(nrand = c(d1.ar1, d2.ar1), dt,
# pad, dj, J1, s0, mother = "morlet")
```
**wtc_sig_parallel**  
*Parallelized Monte Carlo simulation using doParallel package.*

**Description**

Equivalent to *wtc.sig*

**Usage**

```r
wtc_sig_parallel(nrands = 300, lag1, dt, ntimes, pad = TRUE,
                  dj = 1/12, s0, J1, max.scale = NULL, mother = "morlet",
                  sig.level = 0.95, quiet = TRUE)
```

**Arguments**

- **nrands**  
  number of Monte Carlo randomizations. Default is 300.
- **lag1**  
  vector containing the AR(1) coefficient of each time series.
- **dt**  
  length of a time step.
- **ntimes**  
  number of time steps in time series.
- **pad**  
  pad the values will with zeros to increase the speed of the transform. Default is TRUE.
- **dj**  
  spacing between successive scales. Default is 1/12.
- **s0**  
  smallest scale of the wavelet. Default is 2*dt
- **J1**  
  number of scales - 1.
- **max.scale**  
  maximum scale
- **mother**  
  type of mother wavelet function to use. Can be set to *morlet*, *dog*, or *paul*. Default is *morlet*. Significance testing is only available for *morlet* wavelet.
- **sig.level**  
  significance level to compute. Default is 0.95
- **quiet**  
  Do not display progress bar. Default is FALSE

**Examples**

```r
# Not run: library(foreach)
# library(doParallel)
# cl <- makeCluster(4, outfile="") # number of cores. Notice 'outfile'
# registerDoParallel(cl)
# wtc_sig_parallel(your parameters go here)
# stopCluster(cl)
```
xwt  

**Compute cross-wavelet**

### Description

Compute cross-wavelet

### Usage

```
xwt(d1, d2, pad = TRUE, dj = 1/12, s0 = 2 * dt, J1 = NULL, 
    max.scale = NULL, mother = "morlet", param = -1, lag1 = NULL, 
    sig.level = 0.95, sig.test = 0)
```

### Arguments

- **d1**: time series 1 in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
- **d2**: time series 2 in matrix format (n rows x 2 columns). The first column should contain the time steps and the second column should contain the values.
- **pad**: pad the values will with zeros to increase the speed of the transform. Default is TRUE.
- **dj**: spacing between successive scales. Default is 1/12.
- **s0**: smallest scale of the wavelet. Default is 2*dt
- **J1**: number of scales - 1.
- **max.scale**: maximum scale. Computed automatically if left unspecified.
- **mother**: type of mother wavelet function to use. Can be set to morlet, dog, or paul. Default is morlet. Significance testing is only available for morlet wavelet.
- **param**: nondimensional parameter specific to the wavelet function.
- **lag1**: vector containing the AR(1) coefficient of each time series.
- **sig.level**: significance level. Default is 0.95.
- **sig.test**: type of significance test. If set to 0, use a regular \( \chi^2 \) test. If set to 1, then perform a time-average test. If set to 2, then do a scale-average test.

### Value

Returns a biwavelet object containing:
- **coi**: matrix containing cone of influence
- **wave**: matrix containing the cross-wavelet transform
- **wave.corr**: matrix containing the bias-corrected cross-wavelet transform using the method described by Veleda et al. (2012)
- **power**: matrix of power
power.corr matrix of bias-corrected cross-wavelet power using the method described by Veleda et al. (2012)

phase matrix of phases

period vector of periods

scale vector of scales

dt length of a time step

t vector of times

xaxis vector of values used to plot xaxis

s0 smallest scale of the wavelet

dj spacing between successive scales

d1.sigma standard deviation of time series 1

d2.sigma standard deviation of time series 2

mother mother wavelet used

type type of biwavelet object created (xwt)

signif matrix containing significance levels

Author(s)

Tarik C. Gouhier (tarik.gouhier@gmail.com) Code based on WTC MATLAB package written by Aslak Grinsted.

References


Examples

library(biwavelet)

t1 <- cbind(1:100, rnorm(100))
t2 <- cbind(1:100, rnorm(100))

# Compute Cross-wavelet
xwt.tlt2 <- xwt(t1, t2)
plot(xwt.tlt2, plot.cb = TRUE, plot.phase = TRUE,
     main = "Plot cross-wavelet and phase difference (arrows)")
# Real data
data(enviro.data)

# Cross-wavelet of MEI and NPGO
xwt.mei.npgo <- xwt(subset(enviro.data, select = c("date", "mei")),
                   subset(enviro.data, select = c("date", "npgo")))

# Make room to the right for the color bar
par(oma = c(0, 0, 0, 1), mar = c(5, 4, 4, 5) + 0.1)
plot(xwt.mei.npgo, plot.cb = TRUE, plot.phase = TRUE)
Index

*Topic coherence
biwavelet-package, 2

*Topic **cross-wavelet**
biwavelet-package, 2

*Topic datasets
MOTHERS, 11

*Topic dataset
enviro.data, 9

*Topic wavelet
biwavelet-package, 2

ar1.spectrum, 4
ar1_ma0_sim, 5
arrow, 6

biwavelet (biwavelet-package), 2
biwavelet-package, 2

check.data, 6
check.datum, 7
convolve2D, 8, 9
convolve2D_typeopen, 9

enviro.data, 9

get_minroots, 10

image.plot, 14

MOTHERS, 11

phase.plot, 11
plot.biwavelet, 12
pwtc, 15

rcpp_row_quantile, 17
rcpp_wt_bases_dog, 18
rcpp_wt_bases_morlet, 19
rcpp_wt_bases_paul, 20

smooth.wavelet, 9, 21

wt, 24
wt.bases, 26
wt.bases.dog, 26, 27
wt.bases.morlet, 26, 27
wt.bases.paul, 26, 28
wt.sig, 29
wtc, 30
wtc.sig, 32
wtc_sig_parallel, 34

xwt, 35

wclust, 22
wdist, 23

38