Package ‘EMD’

November 28, 2008

Title Empirical Mode Decomposition and Hilbert Spectral Analysis

Version 1.2.0

Date 2008-11-08

Author Donghoh Kim <donghoh.kim@gmail.com>, Hee-Seok Oh <heeseok@stats.snu.ac.kr>

Maintainer Donghoh Kim <donghoh.kim@gmail.com>

Depends R (>= 2.6), fields (>= 4.1)

Description This package carries out empirical mode decompostion and Hilbert spectral analysis.

License GPL version 2 or newer

URL http://dasan.sejong.ac.kr/~dhkim/software_emd.html

R topics documented:

EMD-internal 2
cvtype 2
emd 3
emd.pred 4
emd2d 5
emddenoise 6
extractimf 8
extractimf2d 9
extrema 10
extrema2dC 11
hilbertspec 12
imageEMD 13
kospi200 14
lena 14
lennon 15
solar 15
spectrogram 16
sunspot 17

EMD-internal *Internal EMD Functions*

Description

The `hilbert`, `unwrap` and `image.plot.ts` are internal EMD functions.

Details

These are not to be called by the user.

cvtype *Generating test dataset index for cross-validation*

Description

This function generates test dataset index for cross-validation.

Usage

```
cvtype(n, cv.bsize=1, cv.kfold, cv.random=TRUE)
```

Arguments

`n`	the number of observation
`cv.bsize`	block size of cross-validation
`cv.kfold`	the number of fold of cross-validation
`cv.random`	whether or not random cross-validation scheme should be used. Set `cv.random=TRUE` for random cross-validation scheme

Details

This function provides index of test dataset according to various cross-validation scheme. One may construct K test datasets in a way that each testset consists of blocks of b consecutive data. Set `cv.bsize = b` for this. To select each fold at random, set `cv.random = TRUE`.

Value

matrix of which row is test dataset index for cross-validation.

Examples

```
# Traditional 4-fold cross-validation for 100 observations
cvtype(n=100, cv.bsize=1, cv.kfold=4, cv.random=FALSE)
# Random 4-fold cross-validation with block size 2 for 100 observations
cvtype(n=100, cv.bsize=2, cv.kfold=4, cv.random=TRUE)
```

`emd` *Empirical Mode Decomposition*

Description

This function performs empirical mode decomposition.

Usage

```
emd(xt, tt=NULL, tol=sd(xt)*0.1^2, max.sift=20, stoprule="type1",
    boundary="periodic", smlevels=c(1), sm="none", spar=NA, weight=20,
    check=FALSE, max.imf=10, plot.imf=TRUE, interm=NULL)
```

Arguments

`xt`	observation or signal observed at time `tt`
`tt`	observation index or time index
`tol`	tolerance for stopping rule of sifting
`max.sift`	the maximum number of sifting
`stoprule`	stopping rule of sifting
`boundary`	specifies boundary condition from “none", “wave", “symmetric", “periodic" or “evenodd".
`smlevels`	specifies which level of the IMF is obtained by smoothing other than interpolation.
`sm`	specifies whether envelop is constructed by smoothing spline.
`spar`	specifies user-supplied smoothing parameter of spline.
`weight`	the smoothness of spline is determined by `weight` times smoothing parameter of GCV.
`check`	specifies whether the sifting process is displayed. If `check=TRUE`, click the plotting area to start the next step.
`max.imf`	the maximum number of IMF’s
`plot.imf`	specifies whether each IMF is displayed. If `plot.imf=TRUE`, click the plotting area to start the next step.
`interm`	specifies vector of periods to be excluded from the IMF’s.

Details

This function performs empirical mode decomposition.

Value

`imf`	IMF’s
`residue`	residue signal after extracting IMF’s from observations `xt`
`nimf`	the number of IMF’s

References

Huang, N. E., Shen, Z., Long, S. R., Wu, M. L. Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C. and Liu, H. H. (1998) The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proceedings of the Royal Society London A*, **454**, 903–995.

See Also

`extrema`, `extractimf`.

Examples

```
### Empirical Mode Decomposition
ndata <- 3000
tt2 <- seq(0, 9, length=ndata)
xt2 <- sin(pi * tt2) + sin(2* pi * tt2) + sin(6 * pi * tt2)  + 0.5 * tt2

par(mfrow=c(3,1), mar=c(2,1,2,1))
try <- emd(xt2, tt2, boundary="wave")

### Ploting the IMF's
par(mfrow=c(3,1), mar=c(2,1,2,1))
X11(); par(mfrow=c(try$nimf+1, 1), mar=c(2,1,2,1))
rangeimf <- range(try$imf)
for(i in 1:try$nimf) {
    plot(tt2, try$imf[,i], type="l", xlab="", ylab="", ylim=rangeimf,
    main=paste(i, "-th IMF", sep="")); abline(h=0)
}
plot(tt2, try$residue, xlab="", ylab="", main="residue", type="l",
axes=FALSE); box()
```

`emd.pred` *Prediction by EMD and VAR model*

Description

This function calculates prediction values and confidence limits using EMD and VAR model.

Usage

```
emd.pred(varpred, trendpred, ci = 0.95, figure = TRUE)
```

Arguments

`varpred`	prediction result of IMF's by VAR model.
`trendpred`	prediction result of residue by polynomial regression model.
`ci`	confidence interval level.
`figure`	specifies whether prediction result is displayed.

Details

This function calculates prediction values and confidence limits using EMD and VAR model.

Value

`fcst`	prediction values
`lower`	lower limits of prediction
`upper`	upper limits of prediction

`emd2d` *Two dimensional Empirical Mode Decomposition*

Description

This function performs two dimensional empirical mode decomposition.

Usage

```
emd2d(z, x = NULL, y = NULL, tol = sd(c(z)) * 0.1^2, max.sift = 20,
    boundary = "reflexive", boundperc = 0.3, max.imf = 5,
    smlevels = 1, weight = 0, plot.imf = FALSE)
```

Arguments

`z`	matrix of an image observed at (`x`, `y`)
`x, y`	locations of regular grid at which the values in `z` are measured
`tol`	tolerance for stopping rule of sifting
`max.sift`	the maximum number of sifting
`boundary`	specifies boundary condition from “none", “symmetric" or “reflexive".
`boundperc`	expand an image by adding specified percentage of image at the boundary when boundary condition is ’symmetric’ or ’reflexive’.
`max.imf`	the maximum number of IMF’s
`smlevels`	specifies which level of the IMF is obtained by smoothing other than interpolation.
`weight`	the smoothness of a thin plate spline is determined by `weight` times smoothing parameter of GCV.
`plot.imf`	specifies whether each IMF is displayed. If `plot.imf=TRUE`, click the plotting area to start the next step.

Details

This function performs two dimensional empirical mode decomposition.

Value

`imf`	two dimensional IMF’s
`residue`	residue image after extracting the IMF’s
`maxindex`	index of maxima
`minindex`	index of minima
`nimf`	number of IMF’s

References

Huang, N. E., Shen, Z., Long, S. R., Wu, M. L. Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C. and Liu, H. H. (1998) The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proceedings of the Royal Society London A*, **454**, 903–995.

See Also

`extrema2dC`, `extractimf2d`.

Examples

```
data(lena)
z <- lena[seq(1, 512, by=4), seq(1, 512, by=4)]
image(z, main="Lena", xlab="", ylab="", col=gray(0:100/100), axes=FALSE)
#lenadecom <- emd2d(z, max.imf = 4)
#imageEMD(z=z, emdz=lenadecom, extrema=TRUE, col=gray(0:100/100))
```

`emddenoise` *Denoising by EMD and Cross-Validation*

Description

This function performs denoising by empirical mode decomposition and cross-validation.

Usage

```
emddenoise(xt, tt = NULL, cv.index, cv.level, cv.tol = 0.1^3,
    cv.maxiter = 20, by.imf = FALSE, emd.tol = sd(xt) * 0.1^2,
    max.sift = 20, stoprule = "type1", boundary = "periodic",
    smlevels = c(1), sm = "none", spar = NA, weight = 20,
    check = FALSE, max.imf = 10, plot.imf = FALSE, interm = NULL)
```

Arguments

`xt`	observation or signal observed at time `tt`
`tt`	observation index or time index
`cv.index`	test dataset index according to cross-validation scheme
`cv.level`	levels to be thresholded
`cv.tol`	tolerance for the optimization step of cross-validation
`cv.maxiter`	maximum iteration for the optimization step of cross-validation
`by.imf`	specifies whether shrinkage is performed by each IMS or not.
`emd.tol`	tolerance for stopping rule of sifting
`max.sift`	the maximum number of sifting
`stoprule`	stopping rule of sifting
`boundary`	specifies boundary condition from “none", “wave", “symmetric", “periodic" or “evenodd".
`smlevels`	specifies which level of the IMF is obtained by smoothing other than interpolation.

`sm`	specifies whether envelop is constructed by smoothing spline.
`spar`	specifies user-supplied smoothing parameter of spline.
`weight`	the smoothness of spline is determined by `weight` times smoothing parameter of GCV.
`check`	specifies whether the sifting process is displayed.
`max.imf`	the maximum number of IMF's
`plot.imf`	specifies whether each IMF is displayed.
`interm`	specifies vector of periods to be excluded from the IMF's.

Details

This function performs denoising by empirical mode decomposition and cross-validation.

Value

`dxt`	denoised signal
`optlambda`	threshold values by cross-validation
`lambdaconv`	sequence of lambda's by cross-validation
`perr`	sequence of prediction error by cross-validation
`demd`	denoised IMF's and residue
`niter`	the number of iteration for optimal threshold value

References

Huang, N. E., Shen, Z., Long, S. R., Wu, M. L. Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C. and Liu, H. H. (1998) The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proceedings of the Royal Society London A*, **454**, 903–995.

See Also

`cvtype`, `emd`.

Examples

```
ndata <- 1024
tt <- seq(0, 9, length=ndata)
meanf <- (sin(pi*tt) + sin(2*pi*tt) + sin(6*pi*tt)) * (0.0<tt & tt<=3.0) +
 (sin(pi*tt) + sin(6*pi*tt)) * (3.0<tt & tt<=6.0) +
 (sin(pi*tt) + sin(6*pi*tt) + sin(12*pi*tt)) * (6.0<tt & tt<=9.0)
snr <- 3.0
sigma <- c(sd(meanf[tt<=3]) / snr, sd(meanf[tt<=6 & tt>3]) / snr,
sd(meanf[tt>6]) / snr)
set.seed(1)
error <- c(rnorm(sum(tt<=3), 0, sigma[1]),
rnorm(sum(tt<=6 & tt>3), 0, sigma[2]), rnorm(sum(tt>6), 0, sigma[3]))
xt <- meanf + error

cv.index <- cvtype(n=ndata, cv.kfold=2, cv.random=FALSE)$cv.index
#try10 <- emddenoise(xt, cv.index=cv.index, cv.level=2, by.imf=TRUE)
#try10$optlambda
```

`extractimf` *Intrinsic Mode Function*

Description

This function extracts intrinsic mode function from given a signal.

Usage

```
extractimf(residue, tt=NULL, tol=sd(residue)*0.1^2, max.sift=20,
    stoprule="type1", boundary="periodic", sm="none", spar=NA,
    weight=20, check=FALSE)
```

Arguments

`residue`	observation or signal observed at time `tt`
`tt`	observation index or time index
`tol`	tolerance for stopping rule of sifting
`max.sift`	the maximum number of sifting
`stoprule`	stopping rule of sifting
`boundary`	specifies boundary condition from “none", “wave", “symmetric", “periodic" or “evenodd".
`sm`	specifies whether envelop is constructed by smoothing spline.
`spar`	specifies user-supplied smoothing parameter of spline.
`weight`	the smoothness of spline is determined by `weight` times smoothing parameter of GCV.
`check`	specifies whether the sifting process is displayed. If `check=TRUE`, click the plotting area to start the next step.

Details

This function extracts intrinsic mode function from given a signal.

Value

`imf`	imf
`residue`	residue signal after extracting the finest imf from `residue`
`niter`	the number of iteration to obtain the `imf`

References

Huang, N. E., Shen, Z., Long, S. R., Wu, M. L. Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C. and Liu, H. H. (1998) The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proceedings of the Royal Society London A*, **454**, 903–995.

See Also

`extrema`, `emd`.

Examples

```
### Generating a signal
ndata <- 3000
X11(); par(mfrow=c(1,1), mar=c(1,1,1,1))
tt2 <- seq(0, 9, length=ndata)
xt2 <- sin(pi * tt2) + sin(2* pi * tt2) + sin(6 * pi * tt2)  + 0.5 * tt2
plot(tt2, xt2, xlab="", ylab="", type="l", axes=FALSE); box()

### Extracting the first IMF by sifting process
tryimf <- extractimf(xt2, tt2, check=FALSE)
```

---

extractimf2d *Two dimensional Intrinsic Mode Function*

Description

This function extracts two dimensional intrinsic mode function from given an image.

Usage

```
extractimf2d(residue, x=NULL, y=NULL, nnrow=nrow(residue),
    nncol=ncol(residue), tol=sd(c(residue))*0.1^2,
    max.sift=20, boundary="reflexive", boundperc=0.3, weight=0,
    check=FALSE)
```

Arguments

`residue`	matrix of an image observed at (`x`, `y`)
`x, y`	locations of regular grid at which the values in `residue` are measured
`nnrow`	the number of row of an input image
`nncol`	the number of column of an input image
`tol`	tolerance for stopping rule of sifting
`max.sift`	the maximum number of sifting
`boundary`	specifies boundary condition from “none", “symmetric" or “reflexive".
`boundperc`	expand an image by adding specified percentage of image at the boundary when boundary condition is ’symmetric’ or ’reflexive’.
`weight`	the smoothness of thin plate spline is determined by `weight` times smoothing parameter of GCV.
`check`	specifies whether the sifting process is displayed. If `check=TRUE`, click the plotting area to start the next step.

Details

This function extracts two dimensional intrinsic mode function from given image. For sifting procee, thin plate spline is used.

Value

`imf`	two dimensional IMF
`residue`	residue signal after extracting the finest IMF from `residue`
`maxindex`	index of maxima
`minindex`	index of minima
`niter`	number of iteration obtaining the IMF

References

Huang, N. E., Shen, Z., Long, S. R., Wu, M. L. Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C. and Liu, H. H. (1998) The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proceedings of the Royal Society London A*, **454**, 903–995.

See Also

`extrema2dC`, `emd2d`.

Examples

```
data(lena)
z <- lena[seq(1, 512, by=4), seq(1, 512, by=4)]

#lenaimf1 <- extractimf2d(z, check=FALSE)
```

`extrema` *Finding Local Extrema and Zero-crossings*

Description

This function indentifies extrema and zero-crossings.

Usage

```
extrema(y, ndata = length(y), ndatam1 = ndata - 1)
```

Arguments

`y`	input signal
`ndata`	the number of observation
`ndatam1`	the number of observation - 1

Details

This function indentifies extrema and zero-crossings.

Value

minindex	matrix of time index at which local minima are attained. Each row specifies a starting and ending time index of a local minimum
maxindex	matrix of time index at which local maxima are attained. Each row specifies a starting and ending time index of a local maximum.
nextreme	the number of extrema
cross	matrix of time index of zero-crossings. Each row specifies a starting and ending time index of zero-crossings.
ncross	the number of zero-crossings

See Also

extrema2dC, extractimf, emd.

Examples

```
y <- c(0, 1, 2, 1, -1, 1:4, 5, 6, 0, -4, -6, -5:5, -2:2)
#y <- c(0, 0, 0, 1, -1, 1:4, 4, 4, 0, 0, 0, -5:5, -2:2, 2, 2)
#y <- c(0, 0, 0, 1, -1, 1:4, 4, 4, 0, 0, 0, -5:5, -2:2, 0, 0)

plot(y, type = "b"); abline(h = 0)
extrema(y)
```

---

extrema2dC *Finding Local Extrema*

Description

This function finds the two dimensional local extrema.

Usage

```
extrema2dC(z, nnrow=nrow(z), nncol=ncol(z))
```

Arguments

z	matrix of an input image
nnrow	the number of row of an input image
nncol	the number of column of an input image

Details

This function finds the two dimensional local extrema using Rem's algorithm

Value

minindex	index of minima. Each row specifies index of local minimum.
maxindex	index of maxima. Each row specifies index of local maximum.

See Also

extrema, , extractimf2d, emd2d.

Examples

```
data(lena)
z <- lena[seq(1, 512, by=4), seq(1, 512, by=4)]

par(mfrow=c(1,3), mar=c(0, 0.5, 2, 0.5))
image(z, main="Lena", xlab="", ylab="", col=gray(0:100/100), axes=FALSE)

#example <- extrema2dC(z=z)
#localmin <- matrix(256, 128, 128)
#localmin[example$minindex] <- z[example$minindex]
#image(localmin, main="Local minimum", xlab="", ylab="",
#col=gray(0:100/100), axes=FALSE)

#localmax <- matrix(0, 128, 128)
#localmax[example$maxindex] <- z[example$maxindex]
#image(localmax, main="Local maximum", xlab="", ylab="",
#col=gray(0:100/100), axes=FALSE)
```

---

hilbertspec — *Hilbert Transform and Instantaneous Frequency*

Description

This function calculates teh amplitude and instantaneous frequency using Hilbert transforma.

Usage

```
hilbertspec(xt, tt = NULL)
```

Arguments

xt	matrix of multiple signals. Each column represents a signal.
tt	observation index or time index

Details

This function calculates teh amplitude and instantaneous frequency using Hilbert transforma.

Value

amplitude	matrix of amplitudes for multiple signals xt
instantfreq	matrix of instantaneous frequencies for multiple signals xt
energy	cumulative energy of multiple signals

References

Huang, N. E., Shen, Z., Long, S. R., Wu, M. L. Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C. and Liu, H. H. (1998) The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proceedings of the Royal Society London A*, **454**, 903–995.

See Also

spectrogram.

Examples

```
tt <- seq(0, 0.1, length = 2001)[1:2000]
f1 <- 1776; f2 <- 1000
xt <- sin(2*pi*f1*tt) * (tt <= 0.033 | tt >= 0.067) + sin(2*pi*f2*tt)

### Before treating intermittence
interm1 <- emd(xt, tt, boundary="wave", max.imf=2, plot.imf=FALSE)
### After treating intermittence
interm2 <- emd(xt, tt, boundary="wave", max.imf=2, plot.imf=FALSE,
interm=0.0007)

par(mfrow=c(2,1), mar=c(2,2,2,1))
test1 <- hilbertspec(interm1$imf)
spectrogram(test1$amplitude[,1], test1$instantfreq[,1])

test2 <- hilbertspec(interm2$imf, tt=tt)
spectrogram(test2$amplitude[,1], test2$instantfreq[,1])
```

---

imageEMD *Plot of Two Dimensional Decomposition Result*

Description

This function performs plotting of input image, IMF's, residue and extrema.

Usage

```
imageEMD(z = z, emdz, extrema = FALSE, ...)
```

Arguments

z	matrix of an image
emdz	decomposition result
extrema	specifies whether the extrma is displayed according to the level of IMF
...	the usual arguments to the image function

Details

This function performs plotting of input image, IMF's, residue and extrema.

Examples

```
data(lena)
z <- lena[seq(1, 512, by=4), seq(1, 512, by=4)]
image(z, main="Lena", xlab="", ylab="", col=gray(0:100/100), axes=FALSE)
#lenadecom <- emd2d(z, max.imf = 4)
#imageEMD(z=z, emdz=lenadecom, extrema=TRUE, col=gray(0:100/100))
```

kospi200 *Korea Stock Price Index 200*

Description

the weekly KOSPI 200 index from January, 1990 to February, 2007.

Usage

```
data(kospi200)
```

Format

A list of date and KOSPI200 index

Examples

```
data(kospi200)
names(kospi200)
plot(kospi200$date, kospi200$index, type="l")
```

lena *Gray Lena image*

Description

A 512x512 gray image of Lena.

Usage

```
data(lena)
```

Format

A 512x512 matrix.

Examples

```
data(lena)
image(lena, col=gray(0:100/100), axes=FALSE)
```

lennon — *Gray John Lennon image*

Description

A 256x256 gray image of John Lennon.

Usage

```
data(lennon)
```

Format

A 256x256 matrix.

Examples

```
data(lennon)
image(lennon, col=gray(100:0/100), axes=FALSE)
```

solar — *Solar Irradiance Proxy Data*

Description

solar irradiance proxy data.

Hoyt and Schatten (1993) reconstructed solar irradiance (from 1700 through 1997) using the amplitude of the 11-year solar cycle together with a long term trend estimated from solar-like stars. They put relatively more weight on the length of the 11-year cycle.

Lean et al. (1995) reconstructed solar irradiance (from 1610 through 2000) using the amplitude of the 11-year solar cycle and a long term trend estimated from solar-like stars.

10-Beryllium (10Be) is measured in polar ice from 1424 through 1985. 10-Beryllium (10Be) is produced in the atmosphere by incoming cosmic ray flux, which in turn is influenced by the solar activity. The higher the solar activity, the lower the flux of cosmic radiation entering the earth atmosphere and therefore the lower the production rate of 10Be. The short atmospheric lifetime of 10Be of one to two years (Beer et al. 1994) allows the tracking of solar activity changes and offers an alternative way to the sunspot based techniques for the analysis of the amplitude and length of the solar cycle as well as for low frequency variations.

Usage

```
data(solar.hs)
data(solar.lean)
data(beryllium)
```

Format

A list of year and solar (solar irradiance proxy data) for solar.hs and solar.lean A list of year and be (10-Beryllium) for beryllium

References

Beer, J., Baumgartner, S., Dittrich-Hannen, B., Hauenstein, J., Kubik, P., Lukasczyk, C., Mende, W., Stellmacher, R. and Suter, M. (1994) Solar variability traced by cosmogenic isotopes. *In: Pap, J.M., FrQohlich, C., Hudson, H.S., Solanki, S. (Eds.), The Sun as a Variable Star: Solar and Stellar Irradiance Variations*, Cambridge University Press, Cambridge, 291–300.

Beer, J., Mende, W. and Stellmacher, R. (2000) The role of the sun in climate forcing. *Quaternary Science Reviews*, **19**, 403–415.

Hoyt, D. V and, Schatten, K. H. (1993) A discussion of plausible solar irradiance variations, 1700–1992. *Journal of Geophysical Research*, **98 (A11)**, 18,895–18,906.

Lean, J. L., Beer, J. and Bradley, R. S. (1995) Reconstruction of solar irradiance since 1610: Implications for climate change. *Geophysical Research Letters*, **22 (23)**, 3195–3198.

Oh, H-S, Ammann, C. M., Naveau, P., Nychka, D. and Otto-Bliesner, B. L. (2003) Multi-resolution time series analysis applied to solar irradiance and climate reconstructions. *Journal of Atmospheric and Solar-Terrestrial Physics*, **65**, 191–201.

Examples

```
data(solar.hs)
names(solar.hs)
plot(solar.hs$year, solar.hs$solar, type="l")

data(solar.lean)
names(solar.lean)
plot(solar.lean$year, solar.lean$solar, type="l")

data(beryllium)
names(beryllium)
plot(beryllium$year, beryllium$be, type="l")
```

---

spectrogram *Spectrogram*

Description

This function produces image of amplitude by time index and instantaneous frequency. The horizontal axis represents time, the vertical axis is instantaneous frequency, and the color of each point in the image represents amplitude of a particular frequency at a particular time.

Usage

```
spectrogram(amplitude, freq, tt = NULL, multi = FALSE,
nlevel = NULL, size = NULL)
```

Arguments

amplitude	vector or matrix of amplitudes for multiple signals
freq	vector or matrix of instantaneous frequencies for multiple signals
tt	observation index or time index
multi	specifies whether spectrograms of multiple signals are separated or not.
nlevel	the number of color levels used in legend strip
size	vector of image size.

Details

This function produces image of amplitude by time index and instantaneous frequency. The horizontal axis represents time, the vertical axis is instantaneous frequency, and the color of each point in the image represents amplitude of a particular frequency at a particular time.

Value

image

References

Huang, N. E., Shen, Z., Long, S. R., Wu, M. L. Shih, H. H., Zheng, Q., Yen, N. C., Tung, C. C. and Liu, H. H. (1998) The empirical mode decomposition and Hilbert spectrum for nonlinear and nonstationary time series analysis. *Proceedings of the Royal Society London A*, **454**, 903–995.

See Also

hilbertspec.

Examples

```
tt <- seq(0, 0.1, length = 2001)[1:2000]
f1 <- 1776; f2 <- 1000
xt <- sin(2*pi*f1*tt) * (tt <= 0.033 | tt >= 0.067) + sin(2*pi*f2*tt)

### Before treating intermittence
interm1 <- emd(xt, tt, boundary="wave", max.imf=2, plot.imf=FALSE)
### After treating intermittence
interm2 <- emd(xt, tt, boundary="wave", max.imf=2, plot.imf=FALSE,
interm=0.0007)

par(mfrow=c(2,1), mar=c(2,2,2,1))
test1 <- hilbertspec(interm1$imf)
spectrogram(test1$amplitude[,1], test1$instantfreq[,1])

test2 <- hilbertspec(interm2$imf, tt=tt)
spectrogram(test2$amplitude[,1], test2$instantfreq[,1])
```

---

sunspot *Sunspot Data*

Description

sunspot from 1610 through 1995.

Usage

```
data(sunspot)
```

Format

A list of year and sunspot

References

Oh, H-S, Ammann, C. M., Naveau, P., Nychka, D. and Otto-Bliesner, B. L. (2003) Multi-resolution time series analysis applied to solar irradiance and climate reconstructions. *Journal of Atmospheric and Solar-Terrestrial Physics*, **65**, 191–201.

Examples

```
data(sunspot)
names(sunspot)
plot(sunspot$year, sunspot$sunspot, type="l")
```