Package ‘EPT’

October 12, 2022

Version 0.7.6

Date 2022-01-05

Title Ensemble Patch Transform, Visualization and Decomposition

Author Donghoh Kim [aut, cre],
Hee-Seok Oh [aut],
Guebin Choi [ctb]

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

Depends R (>= 3.0)

Description For multiscale analysis, this package carries out ensemble patch transform, its visualization and multiscale decomposition. The detailed procedure is described in Kim et al. (2020), and Oh and Kim (2020). D. Kim, G. Choi, H.-S. Oh, Ensemble patch transformation: a flexible framework for decomposition and filtering of signal, EURASIP Journal on Advances in Signal Processing 30 (2020) 1-27 <doi:10.1186/s13634-020-00690-7>. H.-S. Oh, D. Kim, Image decomposition by bidimensional ensemble patch transform, Pattern Recognition Letters 135 (2020) 173-179 <doi:10.1016/j.patrec.2020.03.029>.

License GPL (>= 3)

NeedsCompilation no

Repository CRAN

Date/Publication 2022-01-05 01:10:02 UTC

R topics documented:

---

eptdecomp *Decomposition of a signal by Ensemble Patch Transform*

Description

This function decomposes a signal into frequency component and residue of ensemble patch transform by sifting process.

Usage

```
eptdecomp(tindex = NULL, signal, type = "rectangle", tau,
    process = c("average", "average"), pquantile = c(0, 1), equantile = c(0, 1),
    gamma = 1, boundary = "symmetric", stoprule = "type1",
    tol = sd(signal, na.rm = TRUE) * 0.1^2, maxiter = 10, check = FALSE)
```

Arguments

`tindex`	time index at which a signal is observed. When it is `NULL`, the signal is supposed to be equally spaced.
`signal`	a set of data or a signal observed at time `tindex`.
`type`	patch type of `"rectangle"` or `"oval"`.
`tau`	a size parameter for ensemble patch transform.
`process`	specifies transform types for patch and ensemble processes: `process[1]` for patch process and process[2] for ensemble process. Each process has options of `"average"`, `"median"`, or `"envelope"`. Note that when `process[1]` is `"average"` or `"median"`, `process[2]` must be `"average"` or `"median"`. When `process[1]` is `"envelope"`, lower and upper envelopes are obtained by `pquantile[1]` $\times$ 100%-quantile and `pquantile[2]` $\times$ 100%-quantile of patches, respectively. When `process[2]` is `"envelope"`, ensemble lower and upper envelopes are obtained as `equantile[1]` $\times$ 100%-quantile and `equantile[2]` $\times$ 100%-quantile of lower and upper envelopes of shifted patches, respectively.
`pquantile`	quantiles for lower and upper envelopes of patch transform. When it is `c(0, 1)`, minimum and maximum of a patch are used for lower and upper envelopes, respectively.
`equantile`	quantiles for lower and upper envelopes of ensemble patch transform.
`gamma`	controls the amount of envelope magnitude.
`boundary`	specifies boundary condition from `"symmetric"`, `"periodic"` or `"none"`.
`stoprule`	stopping rule of sifting. The `type1` stopping rule indicates that absolute values of ept (ensemble patch transform) component must be less than the user-specified tolerance level. The `type2` is the stopping rule that the difference between ept components at the current and previous iterative steps must be less than the user-specified tolerance level.
`tol`	tolerance for stopping rule of sifting.

maxiter	the maximum number of sifting.
check	specifies whether the sifting process is displayed. When check = TRUE, ept component and residue for each sifting step are displayed. If check=TRUE, click the plotting area to start the next step.

Details

This function decomposes a signal into frequency component and residue of ensemble patch transform by sifting process for a size parameter.

Value

eptcomp	matrix of ept (ensemble patch transform) component at each sifting step.
FC	frequency component of ensemble patch transform by sifting process.
residue	residue of ensemble patch transform by sifting process.
parameters	a list of input parameters of type, tau, process, pquantile, equantile, gamma, boundary, and output parameter niter, the number of sifting.

See Also

eptransf, meptransf.

Examples

```
#### example : composite of two components having different frequencies
ndata <- 1000
tindex <- seq(0, 1, length=ndata)
comp1 <- cos(90*pi*tindex)
comp2 <- cos(10*pi*tindex)
f <- comp1 + comp2

op <- par(mfrow=c(3,1), mar=c(2,2,2,1))
plot(tindex, f, main="a signal", xlab="", ylab="", type='l')
abline(h=0, lty=3)
plot(tindex, comp1, main="high-frequency component", xlab="", ylab="", type='l')
abline(h=0, lty=3)
plot(tindex, comp2, main="low-frequency component", xlab="", ylab="", type='l')
abline(h=0, lty=3)

#### Decomposition by Ensemble Patch Transform
outdecom <- eptdecomp(signal=f, tau=21, process=c("envelope", "average"), pquantile=c(0, 1))

#### Decomposition result
plot(tindex, f, main="a signal", xlab="", ylab="", type='l'); abline(h=0, lty=3)
plot(outdecom$FC, type='l', main="", xlab="", ylab=""); abline(h=0, lty=3)
title(paste0("high-frequency component, tau=", 21))
lines(comp1, col="red", lty=2, lwd=0.5)
plot(outdecom$residue, type="l", main="", xlab="", ylab=""); abline(h=0, lty=3)
title(paste0("low-frequency component, tau=", 21))
lines(comp2, col="red", lty=2, lwd=0.5)
par(op)
```

eptdecomp2d *Decomposition of an Image by Two-dimensional Ensemble Patch Transform*

Description

This function decomposes an image into frequency component and residue of two-dimensional ensemble patch transform by sifting process.

Usage

```
eptdecomp2d(x = NULL, y = NULL, z, type = "rectangle", tau, theta = 0,
    process = c("average", "average"), pquantile = c(0, 1), equantile = c(0, 1),
    gamma = 1, boundary = "reflexive",
    stoprule = "type2", tol = 0.1^2, maxiter = 10, check = FALSE)
```

Arguments

`x, y`	locations of regular grid at which the values in image `z` are measured. When it is `NULL`, the image is supposed to be equally spaced.
`z`	matrix of an image observed at location `(x, y)`.
`type`	patch type of `"rectangle"` or `"oval"`.
`tau`	a size parameter for two-dimensional ensemble patch transform: `tau[1]` for horizontal size and tau[2] for vertical size of a two-dimensional patch. When `length(tau)` is 1, the horizontal and vertical size are the same.
`theta`	a degree of clockwise rotation of a patch.
`process`	specifies transform types for patch and ensemble processes: `process[1]` for patch process and process[2] for ensemble process. Each process has options of `"average"`, `"median"`, or `"envelope"`. Note that when `process[1]` is `"average"` or `"median"`, `process[2]` must be `"average"` or `"median"`. When `process[1]` is `"envelope"`, lower and upper envelopes are obtained by `pquantile[1]` $\times$ 100%-quantile and `pquantile[2]` $\times$ 100%-quantile of patches, respectively. When `process[2]` is `"envelope"`, ensemble lower and upper envelopes are obtained as `equantile[1]` $\times$ 100%-quantile and `equantile[2]` $\times$ 100%-quantile of lower and upper envelopes of shifted patches, respectively.
`pquantile`	quantiles for lower and upper envelopes of patch transform. When it is `c(0, 1)`, minimum and maximum of a patch are used for lower and upper envelopes, respectively.
`equantile`	quantiles for lower and upper envelopes of ensemble patch transform.
`gamma`	controls the amount of envelope magnitude.
`boundary`	specifies boundary condition from `"reflexive"`, `"periodic"` or `"none"`.
`stoprule`	stopping rule of sifting. The `type1` stopping rule indicates that absolute values of ept (ensemble patch transform) component must be less than the user-specified tolerance level. The `type2` is the stopping rule that the difference

	between ept components at the current and previous iterative steps must be less than the user-specified tolerance level.
`tol`	tolerance for stopping rule of sifting.
`maxiter`	the maximum number of sifting.
`check`	specifies whether the sifting process is displayed. When `check = TRUE`, ept component and residue for each sifting step are displayed. If `check=TRUE`, click the plotting area to start the next step.

Details

This function decomposes an image into frequency component and residue of two-dimensional ensemble patch transform by sifting process for a size parameter.

Value

`eptcomp`	list of ept (ensemble patch transform) component at each sifting step when `check=TRUE`.
`FC`	frequency component of ensemble patch transform by sifting process.
`residue`	residue of ensemble patch transform by sifting process.
`parameters`	a list of input parameters of `type`, `tau`, `theta`, `process`, `pquantile`, `equantile`, `gamma`, `boundary`, and output parameter `niter`, the number of sifting.

See Also

`eptransf2d`, `meptransf2d`.

Examples

```
#### example : composite of two components having different frequencies
nr <- nc <- 128; x <- seq(0, 1, length=nr); y <- seq(0, 1, length=nc)

coscomp1 <- outer(cos(20 * pi * x), cos(20 * pi * y))
coscomp2 <- outer(cos(5* pi * x), cos(5 * pi * y))
cosmeanf <- coscomp1 + coscomp2

op <- par(mfcol=c(3,1), mar=c(0,0.5,2,0.5))
image(cosmeanf, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="a composite image")
image(coscomp1, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="high-frequency component")
image(coscomp2, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="low-frequency component")


#### Decomposition by Ensemble Patch Transform
outcossift <- eptdecomp2d(z=cosmeanf, tau=8)

#### Decomposition Result
op <- par(mfrow=c(2,2), mar=c(2,2,2,1))
image(outcossift$FC, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="Decomposed HF")
persp(outcossift$FC, theta = -30, phi = 45, col = "white", xlab="X", ylab="Y", main="Decomposed HF")
image(outcossift$residue, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="Residue")
```

```
persp(outcossift$residue, theta = -30, phi = 45, col = "white", xlab="X", ylab="Y", main="Residue")

par(op)
```

---

eptmap *Multiscale Visualization of Ensemble Patch Transform of a Signal*

Description

This function displays time-scale representation of ensemble patch transform of a signal for a sequence of size parameters.

Usage

```
eptmap(eptransf, taus = eptransf$parameters$tau, maptype = c("C", "D", "DC", "DD"),
    stat = c("pstat", "Epstat", "pM", "EpM", "psd", "Epsd"),
    der = c("time", "tau"), ncolor = 100, ...)
```

Arguments

`eptransf`	R object of ensemble patch transform by `eptransf()` or `meptransf()`.
`taus`	specifies size parameters for time-scale visualization.
`maptype`	specifies `"C"` for centrality map, `"D"` for dispersion map, `"DC"` for derivative of centrality map and `"DD"` for derivative of dispersion map.
`stat`	`"pstat"` for centrality of patch transform, `"Epstat"` for centrality of ensemble patch transform, `"pM"` for mean envelope of patch transform, `"EpM"` for mean envelope of ensemble patch transform, `"psd"` for standard deviation of patch transform and `"Epsd"` for standard deviation of ensemble patch transform.
`der`	specifies derivative with respect to `"time"` or `"tau"`.
`ncolor`	the number of colors (≥ 1) to be in the palette.
`...`	graphical parameters for image.

Details

This function performs multiscale visualization of ensemble patch transform of a signal for a sequence of size parameters. This function creates images with `heat.colors(ncolor)` colors.

Value

image

See Also

`eptransf`, `meptransf`, `eptplot`.

Examples

```
# a doppler signal
n <- 1000
tindex <- seq(0, 1, length=n)
j <- 5
f <- 10 * sqrt(tindex*(1-tindex)) * sin((2*pi*(1+2^((9-4*j)/5))) / (tindex+2^((9-4*j)/5)))

set.seed(7)
fnoise <- f + 0.4 * rnorm(n)

op <- par(mar=c(2,2,2,1))
plot(f, type="l", , xlab="", ylab="", ylim=range(fnoise))
points(fnoise, cex=0.3)


taus <- seq(4, 64, by=4)

# try1 : Multiscale EPT by average patch transform and average ensemble transform
try1 <- meptransf(tindex=tindex, signal=fnoise, taus=taus, process=c("average", "average"))

par(mfrow=c(2,2))
eptmap(try1, maptype="C", stat="pstat", main="centrality of patch transform")
eptmap(try1, maptype="D", stat="psd", main="standard deviation of patch transform")
eptmap(try1, maptype="C", stat="Epstat", main="centrality of ensemble patch transform")
eptmap(try1, maptype="D", stat="Epsd", main="standard deviation of ensemble patch transform")

eptmap(try1, maptype="DC", stat="Epstat", der="time",
    main="derivative of centrality w.r.t time")
eptmap(try1, maptype="DC", stat="Epstat", der="tau",
    main="derivative of centrality w.r.t tau")
eptmap(try1, maptype="DD", stat="Epsd", der="time",
    main="derivative of standard deviation w.r.t time")
eptmap(try1, maptype="DD", stat="Epsd", der="tau",
    main="derivative of standard deviation w.r.t tau", ncolor=70)

# try2 : Multiscale EPT by envelope patch transform and average ensemble transform
try2 <- meptransf(tindex=tindex, signal=fnoise, taus=taus, process=c("envelope", "average"),
    pquantile=c(0, 1))

eptmap(try2, maptype="C", stat="pM", main="mean envelope of patch transform")
eptmap(try2, maptype="C", stat="EpM", main="mean envelope of ensemble patch transform")
eptmap(try2, maptype="DC", stat="EpM", der="time",
    main="derivative of mean envelope w.r.t time")
eptmap(try2, maptype="DC", stat="EpM", der="tau",
    main="derivative of mean envelope w.r.t time")

par(op)
```

---

eptplot *Plot of Components by Ensemble Patch Transform of a Signal*

Description

This function plots ensemble patch transform of a signal for a sequence of size parameters tau's.

Usage

```
eptplot(eptransf, taus = eptransf$parameters$tau)
```

Arguments

`eptransf`	R object of ensemble patch transform by `eptransf()` or `meptransf()`.
`taus`	specifies size parameters for which ensemble patch transform of a signal is displayed.

Details

This function plots ensemble patch transform of a signal for a sequence of size parameters `taus`.

Value

plot

See Also

`eptransf`, `meptransf`, `eptmap`.

Examples

```
n <- 500
set.seed(1)
x <- c(rnorm(n), arima.sim(list(order = c(1,0,0), ar = 0.9), n = n, sd=sqrt(1-0.9^2)))

taus <- seq(10, 100, by=10)

# eptr1 : Multiscale EPT by average patch transform and average ensemble transform
eptr1 <- meptransf(tindex=1:(2*n), signal=x, taus=taus, process=c("average", "average"),
    boundary="none")
names(eptr1)

op <- par(mfcol=c(4,1), mar=c(4,2,2,0.1))
plot(x, xlab="", type="l", main="signal")

eptplot(eptr1)
eptplot(eptr1, taus=20)
eptplot(eptr1, taus=c(20, 30))
lines(eptr1$Epstat[, 2], col="blue")
lines(eptr1$Epstat[, 3], col="red")

# eptr2 : Multiscale EPT by envelope patch transform and average ensemble transform
eptr2 <- meptransf(tindex=1:(2*n), signal=x, type="oval", taus=taus,
    process=c("envelope", "average"), pquantile=c(0,1), gamma=0.06, boundary="none")
names(eptr2)
```

```
plot(x, xlab="", type="l")
eptplot(eptr2)
eptplot(eptr2, taus=20)
eptplot(eptr2, taus=c(20, 30))
lines(eptr2$EpM[, 2], col="blue")
lines(eptr2$EpM[, 3], col="red")
par(op)
```

---

eptransf *Ensemble Patch Transform of a Signal*

Description

This function performs ensemble patch transform of a signal for a size parameter.

Usage

```
eptransf(tindex = NULL, signal, type = "rectangle", tau,
    process = c("average", "average"), pquantile = c(0, 1), equantile = c(0, 1),
    gamma = 1, boundary = "symmetric")
```

Arguments

`tindex`	time index at which a signal is observed. When it is `NULL`, the signal is supposed to be equally spaced.
`signal`	a set of data or a signal observed at time `tindex`.
`type`	patch type of `"rectangle"` or `"oval"`.
`tau`	size parameter for ensemble patch transform.
`process`	specifies transform types for patch and ensemble processes: `process[1]` for patch process and process[2] for ensemble process. Each process has options of `"average"`, `"median"`, or `"envelope"`. Note that when `process[1]` is `"average"` or `"median"`, `process[2]` must be `"average"` or `"median"`. When `process[1]` is `"envelope"`, lower and upper envelopes are obtained by `pquantile[1]` $\times$ 100%-quantile and `pquantile[2]` $\times$ 100%-quantile of patches, respectively. When `process[2]` is `"envelope"`, ensemble lower and upper envelopes are obtained as `equantile[1]` $\times$ 100%-quantile and `equantile[2]` $\times$ 100%-quantile of lower and upper envelopes of shifted patches, respectively.
`pquantile`	quantiles for lower and upper envelopes of patch transform. When it is `c(0, 1)`, minimum and maximum of a patch are used for lower and upper envelopes, respectively.
`equantile`	quantiles for lower and upper envelopes of ensemble patch transform.
`gamma`	controls the amount of envelope magnitude.
`boundary`	specifies boundary condition from `"symmetric"`, `"periodic"` or `"none"`.

Details

This function performs ensemble patch transform of a signal for a size parameter `tau`, and produces statistics and envelopes for ensemble patch transform. When `process[1]` is `"average"` or `"median"`, outputs related to envelopes are defined as `NULL`. When `process[2]` is `"envelope"`, outputs, `pstat` and `Epstat`, are defined as `NULL`.

Value

`tindex`	time index at which a signal is observed.
`signal`	a set of data or a signal observed at time `tindex`.
`pstat`	centrality of patch transform for size parameter `tau`.
`Epstat`	centrality of ensemble patch transform for size parameter `tau`.
`psd`	standard deviation of patch transform for size parameter `tau`.
`Epsd`	standard deviation of ensemble patch transform for size parameter `tau`.
`pL`	lower envelope of patch transform for size parameter `tau`.
`pU`	upper envelope of patch transform for size parameter `tau`.
`pM`	mean envelope, `(pL + pU) / 2`, of patch transform for size parameter `tau`.
`pR`	distance between lower and upper envelopes, `(pU - pL)`, of patch transform for size parameter `tau`.
`EpL`	lower envelope of ensemble patch transform for size parameter `tau`.
`EpU`	upper envelope of ensemble patch transform for size parameter `tau`.
`EpM`	mean envelope, `(EpL + EpU) / 2`, of ensemble patch transform for size parameter `tau`.
`EpR`	distance between lower and upper envelopes, `(EpU - EpL)`, of ensemble patch transform for size parameter `tau`.
`parameters`	a list of input parameters of `type`, `tau`, `process`, `pquantile`, `equantile`, `gamma`, and `boundary`.
`nlevel`	the number of size parameter `tau`. For `eptransf()` function, `nlevel` is 1.

See Also

`meptransf`, `eptdecomp`.

Examples

```
# a doppler signal
n <- 256
tindex <- seq(0, 1, length=n)
j <- 5
f <- 10 * sqrt(tindex*(1-tindex)) * sin((2*pi*(1+2^((9-4*j)/5))) / (tindex+2^((9-4*j)/5)))
fnoise <- f + 0.4 * rnorm(n)

#### Ensemble statistics
op <- par(mfrow=c(5,3), mar=c(2,2,2,1))
layout(matrix(c(1, 1, 1, 2:13), 5, 3, byrow = TRUE))
```

```
plot(f, main="a doppler signal", xlab="", ylab="", type='l', ylim=range(fnoise))
points(fnoise); abline(h=0, lty=3)

#### Ensemble Patch Transform
taus <- c(5, 10, 20)

out <- list()
for (i in 1:length(taus))
    out[[i]] <- eptransf(signal=fnoise, tau=taus[i], process=c("average", "average"))

for (i in 1:length(taus)) {
    plot(out[[i]]$Epstat, type="l", xlab="", ylab="",
        main=paste0("ensemble average of patch mean, tau=", taus[i]))
    abline(h=0, lty=3)
}

for (i in 1:length(taus))
    plot(out[[i]]$Epsd, type='l', xlab="", ylab="",
         main=paste0("ensemble average of standard deviation, tau=", taus[i]))

out2 <- list()
for (i in 1:length(taus))
    out2[[i]] <- eptransf(signal=fnoise, tau=taus[i], process=c("envelope", "average"))

for (i in 1:length(taus)) {
    plot(out2[[i]]$EpM, type="l", col="red", xlab="", ylab="",
         ylim=range(c(out2[[i]]$EpU,out2[[i]]$EpL)),
         main=paste0("ensemble average of mean envelope, tau=", taus[i]))
    points(fnoise, cex=0.1)
    abline(h=0, lty=3); lines(out2[[i]]$EpU); lines(out2[[i]]$EpL)
}

for (i in 1:length(taus))
    plot(out2[[i]]$EpR, type='l', xlab="", ylab="",
         main=paste0("ensemble average of envelope distance, tau=", taus[i]))

par(op)
```

`eptransf2d` *Two-dimensional Ensemble Patch Transform of an Image*

Description

This function performs two-dimensional ensemble patch transform of an image for a size parameter.

Usage

```
eptransf2d(x = NULL, y = NULL, z, type = "rectangle", tau, theta = 0,
```

```
process = c("average", "average"), pquantile = c(0, 1), equantile = c(0, 1),
gamma = 1, boundary = "reflexive")
```

Arguments

`x, y`	locations of regular grid at which the values in image `z` are measured. When those are `NULL`, the image is supposed to be equally spaced.
`z`	matrix of an image observed at location `(x, y)`.
`type`	patch type of `"rectangle"` or `"oval"`.
`tau`	a size parameter for two-dimensional ensemble patch transform: `tau[1]` for horizontal size and tau[2] for vertical size of a two-dimensional patch. When `length(tau)` is 1, the horizontal and vertical size are the same.
`theta`	a degree of clockwise rotation of a patch.
`process`	specifies transform types for patch and ensemble processes: `process[1]` for patch process and process[2] for ensemble process. Each process has options of `"average"`, `"median"`, or `"envelope"`. Note that when `process[1]` is `"average"` or `"median"`, `process[2]` must be `"average"` or `"median"`. When `process[1]` is `"envelope"`, lower and upper envelopes are obtained by `pquantile[1]` $\times$ 100%-quantile and `pquantile[2]` $\times$ 100%-quantile of patches, respectively. When `process[2]` is `"envelope"`, ensemble lower and upper envelopes are obtained as `equantile[1]` $\times$ 100%-quantile and `equantile[2]` $\times$ 100%-quantile of lower and upper envelopes of shifted patches, respectively.
`pquantile`	quantiles for lower and upper envelopes of patch transform. When it is `c(0, 1)`, minimum and maximum of a patch are used for lower and upper envelopes, respectively.
`equantile`	quantiles for lower and upper envelopes of ensemble patch transform.
`gamma`	controls the amount of envelope magnitude.
`boundary`	specifies boundary condition from `"reflexive"`, `"periodic"` or `"none"`.

Details

This function performs two-dimensional ensemble patch transform of an image for a size parameter `tau`, and produces statistics and envelopes for two-dimensional ensemble patch transform. When `process[1]` is `"average"` or `"median"`, outputs related to envelopes are defined as `NULL`. When `process[2]` is `"envelope"`, outputs, `pstat` and `Epstat`, are defined as `NULL`.

Value

`x, y`	locations of regular grid at which the values in image `z` are measured. When it is `NULL`, the image is supposed to be equally spaced.
`z`	matrix of an image observed at location `(x, y)`.
`pstat`	centrality of patch transform for size parameter `tau`.
`Epstat`	centrality of ensemble patch transform for size parameter `tau`.
`psd`	standard deviation of patch transform for size parameter `tau`.
`Epsd`	standard deviation of ensemble patch transform for size parameter `tau`.

pL	lower envelope of patch transform for size parameter tau.
pU	upper envelope of patch transform for size parameter tau.
pM	mean envelope, (pL + pU) / 2, of patch transform for size parameter tau.
pR	distance between lower and upper envelopes, (pU - pL), of patch transform for size parameter tau.
EpL	lower envelope of ensemble patch transform for size parameter tau.
EpU	upper envelope of ensemble patch transform for size parameter tau.
EpM	mean envelope, (EpL + EpU) / 2, of ensemble patch transform for size parameter tau.
EpR	distance between lower and upper envelopes, (EpU - EpL), of ensemble patch transform for size parameter tau.
parameters	a list of input parameters of type, tau, theta, process, pquantile, equantile, gamma, and boundary.
nlevel	the number of size parameter tau. For eptransf2d(), nlevel is 1.

See Also

meptransf2d, eptdecomp2d.

Examples

```
#### example : composite of two components having different frequencies
nr <- nc <- 128; x <- seq(0, 1, length=nr); y <- seq(0, 1, length=nc)

coscomp1 <- outer(cos(20 * pi * x), cos(20 * pi * y))
coscomp2 <- outer(cos(5* pi * x), cos(5 * pi * y))
cosmeanf <- coscomp1 + coscomp2

op <- par(mfcol=c(3,1), mar=c(0,0.5,2,0.5))
image(cosmeanf, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="a composite image")
image(coscomp1, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="high-frequency component")
image(coscomp2, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="low-frequency component")


#### Ensemble average of Ensemble Patch Transform
outcos <- eptransf2d(z=cosmeanf, tau=12)
rangez <- range(cosmeanf)

par(mfrow=c(3,1), mar=c(2,2,2,1))
image(outcos$Epstat, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, zlim=rangez,
      main="ensemble average of patch mean")
persp(outcos$Epstat, theta = -30, phi = 45, col = "white", xlab="X", ylab="Y",
      main="ensemble average of patch mean")
image(outcos$Epsd, xlab="", ylab="", col=gray(0:100/100), axes=FALSE,
      main="ensemble average of standard deviation")

#### Ensemble Envelope of Ensemble Patch Transform
outcos2 <- eptransf2d(z=cosmeanf, tau=12, process = c("envelope", "average"))
par(mfrow=c(2,2), mar=c(2,2,2,1))
```

```
image(outcos2$EpL, xlab="", ylab="", col=gray(0:100/100), axes=FALSE,
      main="ensemble average of lower envelope")
image(outcos2$EpU, xlab="", ylab="", col=gray(0:100/100), axes=FALSE,
      main="ensemble average of upper envelope")
image(outcos2$EpM, xlab="", ylab="", col=gray(0:100/100), axes=FALSE,
      main="ensemble average of mean envelope")
image(outcos2$Epsd, xlab="", ylab="", col=gray(0:100/100), axes=FALSE,
      main="ensemble average of standard deviation")

par(op)
```

`localextrema` *Finding Local Extrema and Zero-crossings of a Signal*

Description

This function identifies local extrema and zero-crossings of a signal.

Usage

```
localextrema(y)
```

Arguments

`y`	a set of data or a signal.

Details

This function identifies local extrema and zero-crossings of a signal.

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

`eptransf`, `eptdecomp`.

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)
localextrema(y)

findextrema <- localextrema(y)
points(findextrema$maxindex, y[findextrema$maxindex], pch=16, col="red")
points(findextrema$minindex, y[findextrema$minindex], pch=16, col="blue")
```

LOD *Length of Day Data*

Description

The length-of-day was produced by Gross (2001) from 20 January 1962 to 6 January 2001. The length-of-day (LOD) data was analyzed in Huang et al. (2003).

Usage

```
data(LOD)
```

Format

A list of LOD, YEAR, MONTH and DATE

References

Gross, R. S. (2001) Combinations of Earth orientation measurements: SPACE2000, COMB2000, and POLE2000. JPL Publication 01-2. Jet Propulsion Laboratory, Pasadena, CA.

Huang, N. E., Wu, M. C., Long, S. R., Shen, S., Qu, W., Gloerson, P. and Fan, K. L. (2003) A confidence limit for the empirical mode decomposition and Hilbert spectral analysis. *Proceedings of the Royal Society London A.*, **459**, 2317–2345.

Examples

```
data(LOD)
names(LOD)

xt <- LOD$LOD[LOD$YEAR >= 1981 & LOD$YEAR <= 2000] # From 1981/1/1 to 2000/12/31
xt <- xt/10^4 # measured in millisecond

# EP transform for LOD
outLOD <- eptransf(signal=xt, tau=15, process=c("envelope", "average"), boundary="none")

# outLOD$EpM : candidate of remaining component
```

```
eptplot(outLOD)

op <- par(mfcol=c(3,1), mar=c(2,2,2,1))
plot(xt, type='l', main="LOD", xlab="", ylab="", ylim=range(xt))
plot(xt - outLOD$EpM, type='l', main="candidate of frequency component
    with half month period", xlab="", ylab=""); abline(h=0, lty=3)
plot(outLOD$EpM, type='l', main="candidate of remaining component",
    xlab="", ylab="", ylim=range(xt))


# sifting
LODdecom1 <- eptdecomp(signal=xt, tau=15, process=c("envelope", "average"),
    boundary="none", tol=sd(xt)*0.1^3, maxiter = 30)

# extraction of frequency component with half month period
plot(xt, type='l', main="LOD", xlab="", ylab="", ylim=range(xt))
plot(LODdecom1$FC, type='l', main="frequency component
    with half month period", xlab="", ylab=""); abline(h=0, lty=3)
plot(LODdecom1$residue, type='l', main="remaining component",
    xlab="", ylab="", ylim=range(xt))

# EP transform for remaining signal from LODdecom1
outLOD2 <- eptransf(signal=LODdecom1$residue, tau=30, process=c("envelope", "average"),
    boundary="none")

# outLOD2$EpM : candidate of remaining component for residue signal from LODdecom1
plot(LODdecom1$residue, type='l', main="remaining component from LODdecom1",
    xlab="", ylab="", ylim=range(xt))
plot(LODdecom1$residue - outLOD2$EpM, type='l', main="candidate of frequency component
    with one month period", xlab="", ylab=""); abline(h=0, lty=3)
plot(outLOD2$EpM, type='l', main="candidate of remaining component",
    xlab="", ylab="", ylim=range(xt))

# sifting
LODdecom2 <- eptdecomp(signal=LODdecom1$residue, tau=30, process=c("envelope", "average"),
    boundary="none", tol=sd(xt)*0.1^3, maxiter = 50)

# extraction of frequency component with one month period
plot(LODdecom1$residue, type='l', main="remaining component from LODdecom1",
    xlab="", ylab="", ylim=range(xt))
plot(LODdecom2$FC, type='l', main="frequency component with one month period",
    xlab="", ylab=""); abline(h=0, lty=3)
plot(LODdecom2$residue, type='l', main="remaining component", xlab="", ylab="",
    ylim=range(xt))

### Decomposition Result
ttt <- paste(LOD$YEAR, LOD$MONTH, LOD$DATE, sep="/")
ttt <- ttt[LOD$YEAR >= 1981 & LOD$YEAR <= 2000]
ttt <- as.Date(ttt)

att <- as.Date(c("1981/1/1", "1982/1/1", "1983/1/1", "1984/1/1", "1985/1/1", "1986/1/1",
                 "1987/1/1", "1988/1/1", "1989/1/1", "1990/1/1", "1991/1/1", "1992/1/1",
                 "1993/1/1", "1994/1/1", "1995/1/1", "1996/1/1", "1997/1/1", "1998/1/1",
```

```
                "1999/1/1", "2000/1/1", "2001/1/1"))

plot(ttt, xt, type='l', main="LOD", xlab="", ylab="", xaxt = "n")
axis(1, at=att, labels=seq(1981, 2001))

plot(ttt, LODdecom1$FC, type="l", main="component with half month period by EPT",
    xlab="", ylab="", xaxt = "n")
axis(1, at=att, labels=seq(1981, 2001)); abline(h=0, lty=3)

plot(ttt, LODdecom2$FC, type="l", main="component with one month period by EPT",
    xlab="", ylab="", xaxt = "n")
axis(1, at=att, labels=seq(1981, 2001)); abline(h=0, lty=3)

par(op)
```

`meptransf` *Multiscale Ensemble Patch Transforms of a Signal*

Description

This function performs multiscale ensemble patch transforms of a signal for a sequence of size parameters.

Usage

```
meptransf(tindex = NULL, signal, type = "rectangle", taus,
    process = c("average", "average"), pquantile = c(0, 1), equantile = c(0, 1),
    gamma = 1, boundary = "symmetric")
```

Arguments

`tindex`	time index at which a signal is observed. When it is `NULL`, the signal is supposed to be equally spaced.
`signal`	a set of data or a signal observed at time `tindex`.
`type`	patch type of `"rectangle"` or `"oval"`.
`taus`	a sequence of size parameters for ensemble patch transform.
`process`	specifies transform types for patch and ensemble processes: `process[1]` for patch process and process[2] for ensemble process. Each process has options of `"average"`, `"median"`, or `"envelope"`. Note that when `process[1]` is `"average"` or `"median"`, `process[2]` must be `"average"` or `"median"`. When `process[1]` is `"envelope"`, lower and upper envelopes are obtained by `pquantile[1]` $\times$ 100%-quantile and `pquantile[2]` $\times$ 100%-quantile of patches, respectively. When `process[2]` is `"envelope"`, ensemble lower and upper envelopes are obtained as `equantile[1]` $\times$ 100%-quantile and `equantile[2]` $\times$ 100%-quantile of lower and upper envelopes of shifted patches, respectively.

pquantile
: quantiles for lower and upper envelopes of patch transform. When it is `c(0, 1)`, minimum and maximum of a patch are used for lower and upper envelopes, respectively.

equantile
: quantiles for lower and upper envelopes of ensemble patch transform.

gamma
: controls the amount of envelope magnitude.

boundary
: specifies boundary condition from `"symmetric"`, `"periodic"` or `"none"`.

Details

This function performs multiscale ensemble patch transforms of a signal for a sequence of size parameters `taus`, and produces statistics and envelopes for ensemble patch transform. When `process[1]` is `"average"` or `"median"`, outputs related to envelopes are defined as `NULL`. When `process[2]` is `"envelope"`, outputs, `pstat` and `Epstat`, are defined as `NULL`.

Value

tindex
: time index at which a signal is observed.

signal
: a set of data or a signal observed at time `tindex`.

pstat
: matrix of centrality of patch transform for a sequence of size parameters `taus`.

Epstat
: matrix of centrality of ensemble patch transform for a sequence of size parameters `taus`.

psd
: matrix of standard deviation of patch transform for a sequence of size parameters `taus`.

Epsd
: matrix of standard deviation of ensemble patch transform for a sequence of size parameters `taus`.

pL
: matrix of lower envelope of patch transform for a sequence of size parameters `taus`.

pU
: matrix of upper envelope of patch transform for a sequence of size parameters `taus`.

pM
: matrix of mean envelope, `(pL + pU) / 2`, of patch transform for a sequence of size parameters `taus`.

pR
: matrix of distance between lower and upper envelopes, `(pU - pL)`, of patch transform for a sequence of size parameters `taus`.

EpL
: matrix of lower envelope of ensemble patch transform for a sequence of size parameters `taus`.

EpU
: matrix of upper envelope of ensemble patch transform for a sequence of size parameters `taus`.

EpM
: matrix of mean envelope, `(EpL + EpU) / 2`, of ensemble patch transform for a sequence of size parameters `taus`.

EpR
: matrix of distance between lower and upper envelopes, `(EpU - EpL)`, of ensemble patch transform for a sequence of size parameters `taus`.

rho
: vector of correlations between `(signal - ept)` component and `ept` component for a sequence of size parameters `taus`. The `ept` component is component obtained by ensemble patch transform.

parameters	a list of input parameters of type, taus, process, pquantile, equantile, gamma, and boundary.
nlevel	the number of size parameters taus.

See Also

eptransf, eptdecomp.

Examples

```
#### example : composite of two components having different frequencies
ndata <- 1000
tindex <- seq(0, 1, length=ndata)
comp1 <- cos(45*pi*tindex)
comp2 <- cos(6*pi*tindex)
f <- comp1 + comp2

op <- par(mfcol=c(3,1), mar=c(2,2,2,1))
plot(tindex, f, main="a signal", xlab="", ylab="", type='l')
abline(h=0, lty=3)
plot(tindex, comp1, main="high-frequency component", xlab="", ylab="", type='l')
abline(h=0, lty=3)
plot(tindex, comp2, main="low-frequency component", xlab="", ylab="", type='l')
abline(h=0, lty=3)


#### Multiscale Ensemble Patch Transform according to tau's
taus1 <- seq(20, 60, by=2)
outmulti <- meptransf(signal=f, taus=taus1, process=c("envelope", "average"),
    pquantile=c(0, 1))

#### To continue, click the plot in case of "locator(1)".
par(mfrow=c(2,2), mar=c(2,2,2,1))
for (i in 1:length(taus1)) {
  plot(f - outmulti$EpM[,i], type='l', main="", xlab="", ylab=""); abline(h=0, lty=3)
  title(paste0("Remaining component for tau=", taus1[i]))
  lines(comp1, col="red", lty=2, lwd=0.5)
  plot(outmulti$EpM[,i], type="l", main=, xlab="", ylab=""); abline(h=0, lty=3)
  title(paste0("Mean envelope of ensemble patch transform for tau=", taus1[i]))
  lines(comp2, col="red", lty=2, lwd=0.5); locator(1)
}

par(op)
```

---

meptransf2d *Multiscale Two-dimensional Ensemble Patch Transforms of an Image*

Description

This function performs multiscale two-dimensional ensemble patch transforms of an image for a sequence of size parameters.

Usage

```
meptransf2d(x = NULL, y = NULL, z, type = "rectangle", taus, theta = 0,
    process = c("average", "average"), pquantile = c(0, 1), equantile = c(0, 1),
    gamma = 1, boundary = "reflexive")
```

Arguments

x, y	locations of regular grid at which the values in image z are measured. When those are NULL, the image is supposed to be equally spaced.
z	matrix of an image observed at location (x, y).
type	patch type of "rectangle" or "oval".
taus	a matrix or vector of size parameters for two-dimensional ensemble patch transform. When it is a matrix, the first and second columns specify the horizontal and vertical sizes of a two-dimensional patch, respectively. When it is a vector, the horizontal and vertical size of a two-dimensional patch are the same.
theta	a degree of clockwise rotation of a patch.
process	specifies transform types for patch and ensemble processes: process[1] for patch process and process[2] for ensemble process. Each process has options of "average", "median", or "envelope". Note that when process[1] is "average" or "median", process[2] must be "average" or "median". When process[1] is "envelope", lower and upper envelopes are obtained by pquantile[1] $\times$ 100%-quantile and pquantile[2] $\times$ 100%-quantile of patches, respectively. When process[2] is "envelope", ensemble lower and upper envelopes are obtained as equantile[1] $\times$ 100%-quantile and equantile[2] $\times$ 100%-quantile of lower and upper envelopes of shifted patches, respectively.
pquantile	quantiles for lower and upper envelopes of patch transform. When it is c(0, 1), minimum and maximum of a patch are used for lower and upper envelopes, respectively.
equantile	quantiles for lower and upper envelopes of ensemble patch transform.
gamma	controls the amount of envelope magnitude.
boundary	specifies boundary condition from "reflexive", "periodic" or "none".

Details

This function performs multiscale two-dimensional ensemble patch transforms of an image for a sequence of size parameters taus, and produces statistics and envelopes for two-dimensional ensemble patch transform. When process[1] is "average" or "median", outputs related to envelopes are defined as NULL. When process[2] is "envelope", outputs, pstat and Epstat, are defined as NULL.

Value

x, y	locations of regular grid at which the values in z are measured. When those are NULL, image is supposed to be equally spaced.
z	matrix of an image observed at (x, y).

pstat	list of centrality of patch transform for a sequence of size parameters taus.
Epstat	list of centrality of ensemble patch transform for a sequence of size parameters taus.
psd	list of standard deviation of patch transform for a sequence of size parameters taus.
Epsd	list of standard deviation of ensemble patch transform for a sequence of size parameters taus.
pL	list of lower envelope of patch transform for a sequence of size parameters taus.
pU	list of upper envelope of patch transform for a sequence of size parameters taus.
pM	list of mean envelope, (pL + pU) / 2, of patch transform for a sequence of size parameters taus.
pR	list of distance between lower and upper envelopes, (pU - pL), of patch transform for a sequence of size parameters taus.
EpL	list of lower envelope of ensemble patch transform for a sequence of size parameters taus.
EpU	list of upper envelope of ensemble patch transform for a sequence of size parameters taus.
EpM	list of mean envelope, (EpL + EpU) / 2, of ensemble patch transform for a sequence of size parameters taus.
EpR	list of distance between lower and upper envelopes, (EpU - EpL), of ensemble patch transform for a sequence of size parameters taus.
rho	vector of correlations between (z - ept) component and ept component for a sequence of size parameters taus. The ept component is component obtained by ensemble patch transform.
parameters	a list of input parameters of type, taus, theta, process, pquantile, equantile, gamma, and boundary.
nlevel	the number of size parameters taus.

See Also

eptransf2d, eptdecomp2d.

Examples

```
#### example : composite of two components having different frequencies
nr <- nc <- 128; x <- seq(0, 1, length=nr); y <- seq(0, 1, length=nc)

coscomp1 <- outer(cos(20 * pi * x), cos(20 * pi * y))
coscomp2 <- outer(cos(5* pi * x), cos(5 * pi * y))
cosmeanf <- coscomp1 + coscomp2

op <- par(mfcol=c(3,1), mar=c(0,0.5,2,0.5))
image(cosmeanf, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="a composite image")
image(coscomp1, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="high-frequency component")
image(coscomp2, xlab="", ylab="", col=gray(0:100/100), axes=FALSE, main="low-frequency component")
```

```
#### Multiscale Ensemble Patch Transform according to tau's
taus1 <- seq(6, 12, by=2)
outcosmulti <- meptransf2d(z=cosmeanf, taus=taus1)

par(mfrow=c(length(taus1), 2), mar=c(2,2,2,1))
for (i in 1:length(taus1)) {
    estlowfreq <- outcosmulti$Epstat[[i]]
    image(estlowfreq, xlab="", ylab="", col=gray(0:100/100), axes=FALSE,
        main=paste0("ensemble average of patch mean, tau=", taus1[i]))
    persp(estlowfreq, theta = -30, phi = 45, col = "white", xlab="X", ylab="Y",
        main=paste0("ensemble average of patch mean, tau=", taus1[i]))
}

par(op)
```

SolarRadiation *Solar Radiation*

Description

The solar radiations were hourly observed at Seoul, Daegu, and Busan in South Korea from September 1, 2003 to September 29, 2003. The data are available from Korea Meteorological Administration (https://data.kma.go.kr). Daegu and Busan, located in the southeast of the Korean Peninsula, are close to each other geographically, whereas Seoul is located in the middle of the Peninsula. In addition, note that Daegu and Busan were severely damaged by a typhoon named “MAEMI" at that time, while Seoul was hardly affected by the typhoon. It is expected that the climatic characteristics of Daegu and Busan are similar, and the pattern of Seoul seems to be different from the other two cities.

Usage

```
data(SolarRadiation)
```

Format

A daraframe of Date, Seoul, Daegu and Busan.

Examples

```
data(SolarRadiation)
names(SolarRadiation)

# ensemble patch transform
SolarRadiationEpU <- SolarRadiationEpL <- NULL

for(j in 1:3) {
   tmp <- eptransf(signal=SolarRadiation[,j+1], tau=24, process=c("envelope", "average"),
        pquantile=c(0, 1), gamma=0)
```

```
    SolarRadiationEpU <- cbind(SolarRadiationEpU, tmp$EpU)
    SolarRadiationEpL <- cbind(SolarRadiationEpL, tmp$EpL)
}

# Correlation of the solar radiations at Seoul, Daegu, and Busan
cor(SolarRadiation[, 2:4])

# Correlation of ensemble average of upper envelope
cor(SolarRadiationEpU)

op <- par(mfrow=c(3,1), mar=c(2,2,2,2))
plot(SolarRadiation[,2], type='l', main="(a) solar-radiation in Seoul and upper envelope",
    ylim=c(0, 3.3), xaxt="n"); axis(1, at=seq(1, 30*24, by=24), labels=seq(1, 30, by=1))
lines(SolarRadiationEpU[,1], lty=2); lines(SolarRadiationEpL[,1], lty=2)
plot(SolarRadiation[,3], type='l', main="(b) solar-radiation in Daegu and upper envelope",
    ylim=c(0, 3.3), xaxt="n"); axis(1, at=seq(1, 30*24, by=24), labels=seq(1, 30, by=1))
lines(SolarRadiationEpU[,2], lty=2); lines(SolarRadiationEpL[,2], lty=2)
plot(SolarRadiation[,4], type='l', main="(c) solar-radiation in Busan and upper envelope",
    ylim=c(0, 3.3), xaxt="n"); axis(1, at=seq(1, 30*24, by=24), labels=seq(1, 30, by=1))
lines(SolarRadiationEpU[,3], lty=2); lines(SolarRadiationEpL[,3], lty=2)
par(op)
```

Index