Package 'mrbsizeR'

May 9, 2026

Type Package

Title Scale Space Multiresolution Analysis of Random Signals

Version 1.3

Date 2024-02-13

Maintainer Roman Flury <roman.flury@math.uzh.ch>

Description A method for the multiresolution analysis of spatial fields and images to capture scale-dependent features.
mrbsizeR is based on scale space smoothing and uses differences of smooths at neighbouring scales for finding features on different scales.
To infer which of the captured features are credible, Bayesian analysis is used.
The scale space multiresolution analysis has three steps: (1) Bayesian signal reconstruction.
(2) Using differences of smooths, scale-
dependent features of the reconstructed signal can be found.
(3) Posterior credibility analysis of the differences of smooths created.
The method has first been proposed by Holmstrom, Pasanen, Furrer, Sain (2011) <DOI:10.1016/j.csda.2011.04.011> and extended in Flury, Gerber, Schmid and Furrer (2021) <DOI:10.1016/j.spasta.2020.100483>.

License GPL-2

Depends R (>= 3.0.0), maps(>= 3.1.1)

Imports fields (>= 8.10), stats (>= 3.0.0), grDevices(>= 3.0.0), graphics(>= 3.0.0), methods(>= 3.0.0), Rcpp (>= 0.12.14)

LinkingTo Rcpp

RoxygenNote 7.2.3

Suggests knitr, rmarkdown, testthat

VignetteBuilder knitr

BugReports https://github.com/romanflury/mrbsizeR/issues

URL https://github.com/romanflury/mrbsizeR, https://romanflury.github.io/mrbsizeR/

NeedsCompilation yes

Author Thimo Schuster [aut],
Roman Flury [cre, aut],
Leena Pasanen [ctb],
Reinhard Furrer [ctb]

Repository CRAN

Date/Publication 2024-02-14 00:01:17 UTC

Contents

`CImap` *Computation of simultaneous credible intervals.*

Description

Simultaneous credible intervals for all differences of smooths at neighboring scales z_i are computed.

Usage

```
CImap(smoothVec, mm, nn, prob = 0.95)
```

Arguments

smoothVec	Differences of smooths at neighboring scales.
mm	Number of rows of the original input object.
nn	Number of columns of the original input object.
prob	Credibility level for the posterior credibility analysis. By default prob = 0.95.

Details

CImap is an internal function of mrbsizeRgrid and is usually not used independently. The output can be analyzed with the plotting function plot.CImapGrid.

Value

An array with simultaneous credible intervals VmapCI and the dimensions of the original input object, mm and nn.

Examples

```
# Artificial sample data: 10 observations (5-by-2 object), 10 samples
set.seed(987)
sampleData  <- matrix(stats::rnorm(100), nrow = 10)
sampleData [4:6, ] <- sampleData [4:6, ] + 5

# Calculation of the simultaneous credible intervals
CImap(smoothVec = sampleData , mm = 5, nn = 2, prob = 0.95)
```

---

dctMatrix *Create a n-by-n discrete cosine transform matrix.*

Description

The discrete cosine transform (DCT) matrix for a given dimension n is calculated.

Usage

```
dctMatrix(n)
```

Arguments

n	Dimension for the DCT matrix.

Details

The function can be used for 1D- or 2D-DCT transforms of data.

- **1D:** Let `Q` be a m-by-n matrix with some data. `D` is a m-by-m DCT matrix created by `dctMatrix(m)`. Then `D %*% Q` returns the discrete cosine transform of the columns of Q. `t(D) %*% Q` returns the inverse DCT of the columns of Q. As D is orthogonal, `solve(D) = t(D)`.
- **2D:** Let `Q` be a m-by-n matrix with some data. `D_m` is a m-by-m DCT matrix created by `dctMatrix(m)`, `D_n` a n-by-n DCT matrix created by `dctMatrix(n)`. `D_m %*% Q %*% t(D_n)` computes the 2D-DCT of Q. The inverse 2D-DCT of Q can be computed via `t(D_mm) %*% DCT_Q %*% D_n`. D_m transforms along columns, D_n along rows. Since D is orthogonal, `solve(D) = t(D)`.

It can be faster to use `dctMatrix` than using a direct transformation, especially when calculating several DCT's.

Value

The n-by-n DCT matrix.

Examples

```
D <- dctMatrix(5)
```

`dftMatrix` *Create a n-by-n discrete Fourier transform matrix.*

Description

The discrete Fourier transform (DFT) matrix for a given dimension n is calculated.

Usage

```
dftMatrix(n)
```

Arguments

`n`	Dimension for the DFT matrix.

Details

The DFT matrix can be used for computing the discrete Fourier transform of a matrix or vector. `dftMatrix(n) %*% testMatrix` is the same as `apply(testMatrix, MARGIN = 2, FUN = fft)`.

Value

The n-by-n DFT matrix.

Examples

```
set.seed(987)
testMatrix <- matrix(sample(1:10, size = 25, replace = TRUE), nrow = 5)
D <- dftMatrix(5)

# Discrete Fourier transform with matrix multiplication:
D %*% testMatrix

# Discrete Fourier transform with function fft:
apply(testMatrix, MARGIN = 2, FUN = fft)
```

`eigenLaplace` *Generate eigenvalues of discrete Laplace matrix.*

Description

The eigenvalues of a discrete Laplace matrix with dimension (`mm`, `nn`) are calculated.

Usage

```
eigenLaplace(mm, nn)
```

Arguments

`mm`	Number of rows of the discrete Laplace matrix.
`nn`	Number of columns of the discrete Laplace matrix.

Value

A row vector containing the eigenvalues of the discrete laplace matrix.

Examples

```
eigval <- eigenLaplace(5, 5)
```

`eigenQsphere` *Generate eigenvalues of precision matrix Q on the surface of a sphere.*

Description

The eigenvalues of the precision matrix Q with dimension (`mm`, `nn`) and polar angle limits `phimin`, `phimax` are calculated.

Usage

```
eigenQsphere(phimin, phimax, mm, nn)
```

Arguments

`phimin`	Polar angle minimum.
`phimax`	Polar angle maximum.
`mm`	Number of rows of precision matrix Q.
`nn`	Number of columns of precision matrix Q.

Details

The corresponding function for data on a grid is `eigenLaplace`.

Value

A list containing 2 elements:

- eigval Row vector containing the eigenvalues of Q.
- eigvec Matrix containing the eigenvectors of Q as columns.

Examples

```
eig_out <- eigenQsphere(phimin = 180/10, phimax = 180 - 180/10, mm = 10, nn = 20)
```

`fftshift` *Swap the quadrants or halves of a 2d matrix.*

Description

`fftshift` is an R equivalent to the Matlab function `fftshift` applied on matrices. For more information about `fftshift` see the Matlab documentation.

Usage

```
fftshift(inputMatrix, dimension = -1)
```

Arguments

`inputMatrix`	Matrix to be swapped.
`dimension`	Which swap should be performed? • `1`: swap halves along the rows. • `2`: swap halves along the columns. • `-1`: swap first quadrant with third and second quadrant with fourth.

Details

It is possible to swap the halves or the quadrants of the input matrix. Halves can be swapped along the rows (`dimension = 1`) or along the columns (`dimension = 2`). When swapping the quadrants, `fftshift` swaps the first quadrant with the third and the second quadrant with the fourth (`dimension = -1`).

Value

Swapped matrix.

Examples

```
set.seed(987)
sampleMat <- matrix(sample(1:10, size = 25, replace = TRUE), nrow = 5)

# Swap halves along the rows:
fftshift(sampleMat, dimension = 1)

# Swap halves along the columns:
fftshift(sampleMat, dimension = 2)

# Swap first quadrant with third and second quadrant with fourth:
fftshift(sampleMat, dimension = -1)
```

HPWmap *Computation of pointwise and highest pointwise probabilities.*

Description

Pointwise (PW) probabilities and highest pointwise (HPW) probabilities of all differences of smooths at neighboring scales are computed.

Usage

```
HPWmap(smoothVec, mm, nn, prob = 0.95)
```

Arguments

smoothVec	Differences of smooths at neighboring scales.
mm	Number of rows of the original input image.
nn	Number of columns of the original input image.
prob	Credibility level for the posterior credibility analysis

Details

HPWmap is an internal function of mrbsizeRgrid and is usually not used independently. The output can be analyzed with the plotting function plot.HPWmapGrid.

Value

List with two arrays:

- pw: Pointwise probabilities (VmapPW) including the dimensions of the original input image, mm and nn.
- hpw: Highest pointwise probabilities (VmapHPW) including the dimensions of the original input image, mm and nn.

Examples

```
# Artificial sample data: 10 observations (5-by-2 object), 10 samples
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, ] <- sampleData[4:6, ] + 5

# Calculation of the simultaneous credible intervals
HPWmap(smoothVec = sampleData, mm = 5, nn = 2, prob = 0.95)
```

`ifftshift` *Inverse FFT shift of a 2d matrix.*

Description

`ifftshift` is an R equivalent to the Matlab function `ifftshift` applied on matrices. For more information about `ifftshift` see the Matlab documentation.

Usage

```
ifftshift(inputMatrix, dimension = -1)
```

Arguments

`inputMatrix`	Matrix to be swapped.
`dimension`	Which swap should be performed? • `1`: swap halves along the rows. • `2`: swap halves along the columns. • `-1`: swap first quadrant with third and second quadrant with fourth.

Details

`ifftshift` is the inverse function to `fftshift`. For more information see the details of `fftshift`

Value

Swapped matrix.

Examples

```
set.seed(987)
sampleMat <- matrix(sample(1:10, size = 25, replace = TRUE), nrow = 5)

# Swap halves along the rows:
ifftshift(sampleMat, dimension = 1)

# Swap halves along the columns:
ifftshift(sampleMat, dimension = 2)

# Swap first quadrant with third and second quadrant with fourth:
ifftshift(sampleMat, dimension = -1)
```

MinLambda *Numerical optimization for finding appropriate smoothing levels.*

Description

Numerical optimization of an objective function G is carried out to find appropriate signal-dependent smoothing levels (λ's). This is easier than visual inspection via the signal-dependent tapering function in TaperingPlot.

Usage

```
MinLambda(Xmu, mm, nn, nGrid, nLambda = 2, lambda, sphere = FALSE)
```

Arguments

Xmu	Posterior mean of the input object as a vector.
mm	Number of rows of the original input object.
nn	Number of columns of the original input object.
nGrid	Size of grid where objective function is evaluated (nGrid-by-nGrid). This argument is ignorded if a sequence lambda is specified.
nLambda	Number of lambdas to minimize over. Possible arguments: 2 (default) or 3.
lambda	λ-sequence which is used for optimization. If nothing is provided, lambda <- 10^seq(-3, 10, len = nGrid) is used for data on a grid and lambda <- 10^seq(-6, 1, len = nGrid) is used for spherical data.
sphere	TRUE or FALSE: Is the input object defined on a sphere?

Details

As signal-dependent tapering functions are quiet irregular, it is hard to find appropriate smoothing values only by visual inspection of the tapering function plot. A more formal approach is the numerical optimization of an objective function.

Optimization can be carried out with 2 or 3 smoothing parameters. As the smoothing parameters 0 and ∞ are always added, this results in a mrbsizeR analysis with 4 or 5 smoothing parameters.

Sometimes, not all features of the input object can be extracted using the smoothing levels proposed by MinLambda. It might then be necessary to include additional smoothing levels.

plot.minLambda creates a plot of the objective function G on a grid. The minimum is indicated with a white point. The minimum values of the λ's can be extracted from the output of MinLambda, see examples.

Value

A list with 3 objects:

G Value of objective function G.

lambda Evaluated smoothing parameters λ.

minind Index of minimal λ's. lambda[minind] gives the minimal values.

Examples

```
# Artificial sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5

# Minimization of two lambdas on a 20-by-20-grid
minlamOut <- MinLambda(Xmu = c(sampleData), mm = 10, nn = 10,
                       nGrid = 20, nLambda = 2)

# Minimal lambda values
minlamOut$lambda[minlamOut$minind]
```

---

mrbsizeR *mrbsizeR: Scale space multiresolution analysis in R.*

Description

mrbsizeR contains a method for the scale space multiresolution analysis of spatial fields and images to capture scale-dependent features. The name is an abbreviation for **M**ulti**R**esolution **B**ayesian **SI**gnificant **ZE**ro crossings of derivatives in **R** and the method combines the concept of statistical scale space analysis with a Bayesian SiZer method.

Details

The mrbsizeR analysis can be applied to data on a regular grid and to spherical data. For data on a grid, the scale space multiresolution analysis has three steps:

1. Bayesian signal reconstruction.
2. Using differences of smooths, scale-dependent features of the reconstructed signal are found.
3. Posterior credibility analysis of the differences of smooths created.

In a first step, Bayesian signal reconstruction is used to extract an underlying signal from a potentially noisy observation. Samples of the resulting posterior can be generated and used for the analysis. For finding features on different scales, differences of smooths at neighboring scales are used. This is an important distinction to other scale space methods (which usually use a wide range of smoothing levels without taking differences) and tries to separate the features into distinct scale categories more aggressively. After a successful extraction of the scale-different features, posterior credibility analysis is necessary to assess whether the features found are "really there" or if they are artifacts of random sampling.

For spherical data, no Bayesian signal reconstruction is implemented in mrbsizer. Data samples therefore need to be available beforehand. The analysis procedure can therefore be summarized in two steps:

1. Using differences of smooths, scale-dependent features of the reconstructed signal are found.
2. Posterior credibility analysis of the differences of smooths created.

This method has first been proposed by Holmstrom, Pasanen, Furrer, Sain (2011), see also http://cc.oulu.fi/~lpasanen/MRBSiZer/.

Major Functions

- `TaperingPlot` Graphical estimation of useful smoothing levels. Can be used signal-independent and signal-dependent.
- `MinLambda` Numerical estimation of useful smoothing levels. Takes the underlying signal into account. `plot.minLambda` can be used for plotting the result.
- `rmvtDCT` Creates samples on a regular grid from a multivariate t_ν-distribution using a discrete cosine transform (DCT).
- `mrbsizeRgrid` Interface of the mrbsizeR method for data on a regular grid. Differences of smooths at neighboring scales are created and posterior credibility analysis is conducted. The results can be visualized using `plot.smMeanGrid`,`plot.HPWmapGrid` and `plot.CImapGrid`.
- `mrbsizeRsphere` Interface of the mrbsizeR method for data on a sphere. Differences of smooths at neighboring scales are created and posterior credibility analysis is conducted. The results can be visualized using `plot.smMeanSphere`,`plot.HPWmapSphere` and `plot.CImapSphere`. For data on a sphere, no Bayesian signal reconstruction is implemented. Samples have to be provided instead.

Getting Started

The vignette for this package offers an extensive overview of the functionality and the usage of `mrbsizeR`.

References

- Holmstrom, L. and Pasanen, L. (2011). MRBSiZer. http://cc.oulu.fi/~lpasanen/MRBSiZer/. Accessed: 2017-03-04.
- Holmstrom, L., Pasanen, L., Furrer, R., and Sain, S. R. (2011). Scale space multiresolution analysis of random signals. Computational Statistics and Data Analysis, 55, 2840-2855. <DOI:10.1016/j.csda.2011.04.011>.
- Holmstrom, L. and Pasanen, L. (2016). Statistical scale space methods. International Statistical Review. <DOI:10.1111/insr.12155>.

DISCLAIMER: The author can not guarantee the correctness of any function or program in this package.

Examples

```
# Artificial sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5

# Generate samples from multivariate t-distribution
tSamp <- rmvtDCT(object = sampleData, lambda = 0.2, sigma = 6, nu0 = 15,
                 ns = 1000)

# mrbsizeRgrid analysis
mrbOut <- mrbsizeRgrid(posteriorFile = tSamp$sample, mm = 10, nn = 10,
```

```
                        lambdaSmoother = c(1, 1000), prob = 0.95)

# Posterior mean of the differences of smooths
plot(x = mrbOut$smMean, turn_out = TRUE)

# Credibility analysis using simultaneous credible intervals
plot(x = mrbOut$ciout, turn_out = TRUE)
```

`mrbsizeRgrid` *Multiresolution analysis of random signals.*

Description

`mrbsizeRgrid` is the interface of the scale space multiresolution method for data on a regular grid. Here, the differences of smooths as well as the posterior credibility analysis are computed. The output can be analyzed with the plotting functions `plot.smMeanGrid`, `plot.CImapGrid` and `plot.HPWmapGrid`.

Usage

```
mrbsizeRgrid(
  posteriorFile,
  mm,
  nn,
  lambdaSmoother,
  prob = 0.95,
  smoothOut = FALSE
)
```

Arguments

`posteriorFile`	Matrix with posterior samples as column vectors.
`mm`	Number of rows of the original object.
`nn`	Number of columns of the original object.
`lambdaSmoother`	Vector consisting of the smoothing levels to be used.
`prob`	Credibility level for the posterior credibility analysis.
`smoothOut`	Should the differences of smooths at neighboring scales be returned as output (FALSE by default)?

Details

`mrbsizeRgrid` conducts two steps of the scale space multiresolution analysis:

1. Extraction of scale-dependent features from the reconstructed signal. This is done by smoothing at different smoothing levels and taking the difference of smooths at neighboring scales.

2. Posterior credibility analysis of the differences of smooths created. Three different methods are applied: Pointwise probabilities (see HPWmap), highest pointwise probabilities (see HPWmap) and simultaneous credible intervals (see CImap).

The signal can be reconstructed using the build-in multivariate t-distribution sampling rmvtDCT. It is also possible to provide samples generated with other methods, see the parameter posteriorFile and the examples.

For further information and examples, see the vignette.

Value

A list containing the following sublists:

smMean Posterior mean of all differences of smooths created.

hpout Pointwise (PW) and highest pointwise (HPW) probabilities of all differences of smooths created.

ciout Simultaneous credible intervals (CI) of all differences of smooths created.

smoothSamples Samples of differences of smooths at neighboring scales, as column vectors.

Examples

```
# Artificial sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5

# Generate samples from multivariate t-distribution
tSamp <- rmvtDCT(object = sampleData, lambda = 0.2, sigma = 6, nu0 = 15,
                 ns = 1000)

# mrbsizeRgrid analysis
mrbOut <- mrbsizeRgrid(posteriorFile = tSamp$sample, mm = 10, nn = 10,
                       lambdaSmoother = c(1, 1000), prob = 0.95)
```

---

mrbsizeRsphere *Multiresolution analysis of random signals for spherical data.*

Description

mrbsizeRSphere is the interface of the scale space multiresolution method for spherical data. Here, the differences of smooths as well as the posterior credibility analysis are computed. The output can be analyzed with the plotting functions plot.smMeanSphere, plot.CImapSphere and plot.HPWmapSphere.

Usage

```
mrbsizeRsphere(
  posteriorFile,
  mm,
  nn,
  lambdaSmoother,
  prob = 0.95,
  smoothOut = FALSE
)
```

Arguments

posteriorFile	Matrix with posterior samples as column vectors.
mm	Number of rows of the original object.
nn	Number of columns of the original object.
lambdaSmoother	Vector consisting of the smoothing levels to be used.
prob	Credibility level for the posterior credibility analysis.
smoothOut	Should the differences of smooths at neighboring scales be returned as output (FALSE by default)?

Details

In contrast to mrbsizeRgrid, mrbsizeRsphere does not conduct Bayesian signal reconstruction via sampling from a posterior distribution. Samples of the posterior distribution have to be provided instead.

For further information and examples, see mrbsizeRgrid and the vignette.

Value

A list containing the following sublists:

smMean Posterior mean of all differences of smooths created.

hpout Pointwise (PW) and highest pointwise (HPW) probabilities of all differences of smooths created.

ciout Simultaneous credible intervals (CI) of all differences of smooths created.

smoothSamples Samples of differences of smooths at neighboring scales, as column vectors.

Examples

```
# Artificial spherical sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(2000), nrow = 200)
sampleData[50:65, ] <- sampleData[50:65, ] + 5

# mrbsizeRsphere analysis
mrbOut <- mrbsizeRsphere(posteriorFile = sampleData, mm = 10, nn = 20,
                         lambdaSmoother = c(1, 1000), prob = 0.95)
```

`plot.CImapGrid` *Plot of simultaneous credible intervals.*

Description

Maps with simultaneous credible intervals for all differences of smooths at neighboring scales z_i are plotted.

Usage

```
## S3 method for class 'CImapGrid'
plot(
  x,
  color = c("firebrick1", "gainsboro", "dodgerblue3"),
  turnOut = TRUE,
  title,
  aspRatio = 1,
  ...
)
```

Arguments

`x`	List containing the simultaneous credible intervals for all differences of smooths.
`color`	Vector of length 3 containing the colors to be used in the credibility maps. The first color represents the credibly negative pixels, the second color the pixels that are not credibly different from zero and the third color the credibly positive pixels.
`turnOut`	Logical. Should the output images be turned 90 degrees counter-clockwise?
`title`	Vector containing one string per plot. The required number of titles is equal to `length(mrbOut$ciout)`. If no `title` is passed, defaults are used.
`aspRatio`	Adjust the aspect ratio of the plots. The default `aspRatio = 1` produces square plots.
`...`	Further graphical parameters can be passed.

Details

The default colors of the maps have the following meaning:

- **Blue**: Credibly positive pixels.
- **Red**: Credibly negative pixels.
- **Grey**: Pixels that are not credibly different from zero.

`x` corresponds to the `ciout`-part of the output of `mrbsizeRgrid`.

Value

Plots of simultaneous credible intervals for all differences of smooths are created.

Examples

```
# Artificial sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5

# Generate samples from multivariate t-distribution
tSamp <- rmvtDCT(object = sampleData, lambda = 0.2, sigma = 6, nu0 = 15,
                  ns = 1000)

# mrbsizeRgrid analysis
mrbOut <- mrbsizeRgrid(posteriorFile = tSamp$sample, mm = 10, nn = 10,
                       lambdaSmoother = c(1, 1000), prob = 0.95)

# Posterior mean of the differences of smooths
plot(x = mrbOut$smMean, turnOut = TRUE)

# Credibility analysis using simultaneous credible intervals
plot(x = mrbOut$ciout, turnOut = TRUE)
```

`plot.CImapSphere` *Plotting of simultaneous credible intervals on a sphere.*

Description

Maps with simultaneous credible intervals for all differences of smooths at neighboring scales z_i are plotted. Continental lines are added.

Usage

```
## S3 method for class 'CImapSphere'
plot(
  x,
  lon,
  lat,
  color = c("firebrick1", "gainsboro", "dodgerblue3"),
  turnOut = FALSE,
  title,
  ...
)
```

Arguments

`x`	List containing the simultaneous credible intervals of all differences of smooths.
`lon`	Vector containing the longitudes of the data points.
`lat`	Vector containing the latitudes of the data points.

color	Vector of length 3 containing the colors to be used in the credibility maps. The first color represents the credibly negative pixels, the second color the pixels that are not credibly different from zero and the third color the credibly positive pixels.
turnOut	Logical. Should the output images be turned 90 degrees counter-clockwise?
title	Vector containing one string per plot. The required number of titles is equal to `length(mrbOut$ciout)`. If no `title` is passed, defaults are used.
...	Further graphical parameters can be passed.

Details

The default colors of the maps have the following meaning:

- **Blue**: Credibly positive pixels.
- **Red**: Credibly negative pixels.
- **Grey**: Pixels that are not credibly different from zero.

`x` corresponds to the `ciout`-part of the output of `mrbsizeRsphere`.

Value

Plots of simultaneous credible intervals for all differences of smooths are created.

Examples

```
# Artificial spherical sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(2000), nrow = 200)
sampleData[50:65, ] <- sampleData[50:65, ] + 5
lon <- seq(-180, 180, length.out = 20)
lat <- seq(-90, 90, length.out = 10)

# mrbsizeRsphere analysis
mrbOut <- mrbsizeRsphere(posteriorFile = sampleData, mm = 20, nn = 10,
                         lambdaSmoother = c(0.1, 1), prob = 0.95)

# Posterior mean of the differences of smooths
plot(x = mrbOut$smMean, lon = lon, lat = lat,
     color = fields::tim.colors())

# Credibility analysis using simultaneous credible intervals
plot(x = mrbOut$ciout, lon = lon, lat = lat)
```

`plot.HPWmapGrid` *Plotting of pointwise and highest pointwise probabilities.*

Description

Maps with pointwise (PW) probabilities and/or highest pointwise (HPW) probabilities of all differences of smooths at neighboring scales are plotted.

Usage

```
## S3 method for class 'HPWmapGrid'
plot(
  x,
  plotWhich = "Both",
  color = c("firebrick1", "gainsboro", "dodgerblue3"),
  turnOut = TRUE,
  title,
  aspRatio = 1,
  ...
)
```

Arguments

`x`	List containing the pointwise (PW) and highest pointwise (HPW) probabilities of all differences of smooths.
`plotWhich`	Which probabilities shall be plotted? `HPW`, `PW` or `Both`?
`color`	Vector of length 3 containing the colors to be used in the credibility maps. The first color represents the credibly negative pixels, the second color the pixels that are not credibly different from zero and the third color the credibly positive pixels.
`turnOut`	Logical. Should the output images be turned 90 degrees counter-clockwise?
`title`	Vector containing one string per plot. The required number of titles is equal to `length(mrbOut$hpout)`. If no `title` is passed, defaults are used.
`aspRatio`	Adjust the aspect ratio of the plots. The default `aspRatio = 1` produces square plots.
`...`	Further graphical parameters can be passed.

Details

The default colors of the maps have the following meaning:

- **Blue**: Credibly positive pixels.
- **Red**: Credibly negative pixels.
- **Grey**: Pixels that are not credibly different from zero.

`x` corresponds to the `hpout`-part of the output of `mrbsizeRgrid`.

Value

Plots of pointwise and/or highest pointwise probabilities for all differences of smooths are created.

Examples

```
# Artificial sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5

# Generate samples from multivariate t-distribution
tSamp <- rmvtDCT(object = sampleData, lambda = 0.2, sigma = 6, nu0 = 15,
                  ns = 1000)

# mrbsizeRgrid analysis
mrbOut <- mrbsizeRgrid(posteriorFile = tSamp$sample, mm = 10, nn = 10,
                       lambdaSmoother = c(1, 1000), prob = 0.95)

# Posterior mean of the differences of smooths
plot(x = mrbOut$smMean, turnOut = TRUE)

# Credibility analysis using pointwise (PW) maps
plot(x = mrbOut$hpout, plotWhich = "PW", turnOut = TRUE)

# Credibility analysis using highest pointwise probability (HPW) maps
plot(x = mrbOut$hpout, plotWhich = "HPW", turnOut = TRUE)
```

---

plot.HPWmapSphere *Plotting of pointwise and highest pointwise probabilities on a sphere.*

Description

Maps with pointwise (PW) probabilities and/or highest pointwise (HPW) probabilities of all differences of smooths at neighboring scales are plotted. Continental lines are added.

Usage

```
## S3 method for class 'HPWmapSphere'
plot(
  x,
  lon,
  lat,
  plotWhich = "Both",
  color = c("firebrick1", "gainsboro", "dodgerblue3"),
  turnOut = FALSE,
  title,
  ...
)
```

Arguments

x	List containing the pointwise (PW) and highest pointwise (HPW) probabilities of all differences of smooths.
lon	Vector containing the longitudes of the data points.
lat	Vector containing the latitudes of the data points.
plotWhich	Which probabilities shall be plotted? HPW, PW or Both?
color	Vector of length 3 containing the colors to be used in the credibility maps. The first color represents the credibly negative pixels, the second color the pixels that are not credibly different from zero and the third color the credibly positive pixels.
turnOut	Logical. Should the output images be turned 90 degrees counter-clockwise?
title	Vector containing one string per plot. The required number of titles is equal to length(mrbOut$hpout). If no title is passed, defaults are used.
...	Further graphical parameters can be passed.

Details

The default colors of the maps have the following meaning:

- **Blue**: Credibly positive pixels.
- **Red**: Credibly negative pixels.
- **Grey**: Pixels that are not credibly different from zero.

x corresponds to the hpout-part of the output of mrbsizeRsphere.

Value

Plots of pointwise and/or highest pointwise probabilities for all differences of smooths are created.

Examples

```
# Artificial spherical sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(2000), nrow = 200)
sampleData[50:65, ] <- sampleData[50:65, ] + 5
lon <- seq(-180, 180, length.out = 20)
lat <- seq(-90, 90, length.out = 10)

# mrbsizeRsphere analysis
mrbOut <- mrbsizeRsphere(posteriorFile = sampleData, mm = 20, nn = 10,
                         lambdaSmoother = c(0.1, 1), prob = 0.95)

# Posterior mean of the differences of smooths
plot(x = mrbOut$smMean, lon = lon, lat = lat,
     color = fields::tim.colors())

# Credibility analysis using pointwise (PW) maps
plot(x = mrbOut$hpout, lon = lon, lat = lat, plotWhich = "PW")
```

```
# Credibility analysis using highest pointwise probability (HPW) maps
plot(x = mrbOut$hpout, lon = lon, lat = lat, plotWhich = "HPW")
```

`plot.minLambda` *Plot of objective function for finding appropriate smoothing parameters.*

Description

The objective function G is plotted on a grid. The minimum is indicated with a white point.

Usage

```
## S3 method for class 'minLambda'
plot(x, ...)
```

Arguments

`x`	Output of function `MinLambda`.
`...`	Further graphical parameters can be passed.

Details

When minimizing over 2 λ's, one plot is generated: (λ_2 vs λ_3). With 3 λ's, 3 plots are generated: λ_2 vs. λ_3, λ_2 vs. λ_4 and λ_3 vs. λ_4.

Value

Plot of G on a grid.

Examples

```
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5

# Minimization of two lambdas on a 20-by-20-grid
minLamOut <- MinLambda(Xmu = c(sampleData), mm = 10, nn = 10,
                       nGrid = 20, nLambda = 3)

# Plot of the objective function
plot(x = minLamOut)
```

`plot.smMeanGrid` *Plotting of scale-dependent features.*

Description

Scale-dependent features are plotted using differences of smooths at neighboring scales. The features are summarized by their posterior mean.

Usage

```
## S3 method for class 'smMeanGrid'
plot(
  x,
  color.pallet = fields::tim.colors(),
  turnOut = TRUE,
  title,
  aspRatio = 1,
  ...
)
```

Arguments

`x`	List containing the posterior mean of all differences of smooths.
`color.pallet`	The color pallet to be used for plotting scale-dependent features.
`turnOut`	Logical. Should the output images be turned 90 degrees counter-clockwise?
`title`	Vector containing one string per plot. The required number of titles is equal to `length(mrbOut$smMean)`. If no `title` is passed, defaults are used.
`aspRatio`	Adjust the aspect ratio of the plots. The default `aspRatio = 1` produces square plots.
`...`	Further graphical parameters can be passed.

Details

`x` corresponds to the `smmean`-part of the output of `mrbsizeRgrid`.

Value

Plots of the differences of smooths are created.

Examples

```
# Artificial sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5
```

```
# Generate samples from multivariate t-distribution
tSamp <- rmvtDCT(object = sampleData, lambda = 0.2, sigma = 6, nu0 = 15,
                  ns = 1000)

# mrbsizeRgrid analysis
mrbOut <- mrbsizeRgrid(posteriorFile = tSamp$sample, mm = 10, nn = 10,
                       lambdaSmoother = c(1, 1000), prob = 0.95)

# Posterior mean of the differences of smooths
plot(x = mrbOut$smMean, turnOut = TRUE)
```

`plot.smMeanSphere` *Plotting of scale-dependent features on a sphere.*

Description

Scale-dependent features are plotted using differences of smooths at neighboring scales. The features are summarized by their posterior mean. Continental lines are added to the plots.

Usage

```
## S3 method for class 'smMeanSphere'
plot(
  x,
  lon,
  lat,
  color.pallet = fields::tim.colors(),
  turnOut = TRUE,
  title,
  ...
)
```

Arguments

`x`	List containing the posterior mean of all differences of smooths.
`lon`	Vector containing the longitudes of the data points.
`lat`	Vector containing the latitudes of the data points.
`color.pallet`	The color pallet to be used for plotting scale-dependent features.
`turnOut`	Logical. Should the output images be turned 90 degrees counter-clockwise?
`title`	Vector containing one string per plot. The required number of titles is equal to `length(mrbOut$smMean)`. If no `title` is passed, defaults are used.
`...`	Further graphical parameters can be passed.

Details

`x` corresponds to the `smmean`-part of the output of `mrbsizeRsphere`.

Value

Plots of the differences of smooths are created.

Examples

```
# Artificial spherical sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(2000), nrow = 200)
sampleData[50:65, ] <- sampleData[50:65, ] + 5
lon <- seq(-180, 180, length.out = 20)
lat <- seq(-90, 90, length.out = 10)

# mrbsizeRsphere analysis
mrbOut <- mrbsizeRsphere(posteriorFile = sampleData, mm = 20, nn = 10,
                         lambdaSmoother = c(0.1, 1), prob = 0.95)

# Posterior mean of the differences of smooths
plot(x = mrbOut$smMean, lon = lon, lat = lat,
     color = fields::tim.colors(), turnOut = FALSE)
```

`rmvtDCT` *Sampling from marginal posterior multivariate t-distribution.*

Description

Samples from a marginal posterior multivariate t-distribution with normal-inverse-chi-squared-prior are generated.

Usage

```
rmvtDCT(object, lambda, sigma, nu0, ns)
```

Arguments

`object`	Observed object, as `matrix`.
`lambda`	Scaling parameter (λ) of the normal-inverse-chi-squared-prior.
`sigma`	Square root of the σ_0^2 parameter of the normal-inverse-chi-squared-prior.
`nu0`	Degrees of freedom (ν_0) of the normal-inverse-chi-square-prior.
`ns`	Number of samples that should be generated.

Details

An eigenvalue decomposition is used for sampling. To speed up computations, a 2D discrete cosine transform (DCT) has been implemented, see `dctMatrix`. The output is a list containing

1. Samples of the marginal posterior of the input as column vectors.
2. The mean of the marginal posterior of the input as a vector.

Value

A list containing the following elements:

`sample` Samples of the marginal posterior of the input.

`mu` Mean of the marginal posterior of the input.

Examples

```
# Artificial sample data
set.seed(987)
sampleData <- matrix(stats::rnorm(100), nrow = 10)
sampleData[4:6, 6:8] <- sampleData[4:6, 6:8] + 5

# Sampling from a multivariate t-distribution
t_dist_samp <- rmvtDCT(object = sampleData, lambda = 1, sigma = 10,
                       nu0 = 50, ns = 1000)
```

TaperingPlot *Plot of tapering functions.*

Description

Tapering functions corresponding to the smoothing levels in `lambdaSmoother` are drawn. This plot helps to assess if the chosen smoothing levels are appropriate.

Usage

```
TaperingPlot(lambdaSmoother, mm, nn, Xmu, returnseq = FALSE, ...)
```

Arguments

`lambdaSmoother`	Vector consisting of the smoothing levels to be used.
`mm`	Number of rows of the original input object.
`nn`	Number of columns of the original input object.
`Xmu`	If availabe, posterior mean of the input object.
`returnseq`	instead of plotting return the tapering sequences.
`...`	Further graphical parameters can be passed.

Details

The tapering functions of the smoothing levels chosen should be generally approximately disjoint. This will produce features which are somewhat orthogonal. With orthogonal features, it is likely that each difference of smooths corresponds to a different pattern in the input image.

Sometimes, not all patterns of the input image can be extracted using smoothing levels whose tapering functions are disjoint. It might then be necessary to include additional smoothing levels, and the disjointedness might not be satisfied anymore. The selection of appropriate smoothing

levels with this method therefore requires some user interaction. Still, choosing disjoint tapering functions for finding appropriate smoothing levels is a good starting point.

Better results could be obtained if the structure of the posterior mean of the input is also taken into account. If the posterior mean is available, it can be added with the argument `Xmu` and moving averages of the absolute values of the signal-dependent tapering functions are drawn. `MinLambda` offers a more formal approach of optimizing the disjointedness of the tapering functions and can help finding appropriate smoothing levels when the input signal is taken into account.

Value

Plots of the tapering functions for all differences of smooths at neighboring scales are created.

Examples

```
# Signal-independent tapering function plot for a 30-by-10 object with
# the smoothing parameter sequence [0, 1, 10, 1000, inf]:

TaperingPlot(lambdaSmoother = c(1, 10, 1000), mm = 30, nn = 10)


# Signal-dependent tapering function plot for a 30-by-10 object with
# the smoothing parameter sequence [0, 1, 10, 1000, inf]:

set.seed(987)
xmuExample <- c(stats::rnorm(300))
TaperingPlot(lambdaSmoother = c(1, 10, 1000), mm = 30, nn = 10,
             Xmu = xmuExample)
```

---

tridiag *Generate a tridiagonal matrix.*

Description

Generate a tridiagonal matrix with `upperDiag` as superdiagonal, `lowerDiag` as subdiagonal and `mainDiag` as diagonal.

Usage

```
tridiag(mainDiag, upperDiag, lowerDiag)
```

Arguments

`mainDiag`	Diagonal of tridiagonal matrix.
`upperDiag`	Superdiagonal of tridiagonal matrix. Must have length `length(mainDiag) - 1`.
`lowerDiag`	Subdiagonal of tridiagonal matrix. Must have length `length(mainDiag) - 1`.

Value

Tridiagonal matrix.

Examples

```
set.seed(987)
mainDiag <- sample(100:110, size = 6, replace = TRUE)
upperDiag <- sample(10:20, size = 5, replace = TRUE)
lowerDiag <- sample(1:10, size = 5, replace = TRUE)

tridiag(mainDiag, upperDiag, lowerDiag)
```

---

turnmat *Turn matrix 90 degrees counter-clockwise.*

Description

Help function to turn matrix 90 degrees counter-clockwise. `turnmat` is used as an internal function in some of the plotting functions. Depending on how the data is stored, it can be necessary to turn the matrices 90 degrees counter-clockwise for being able to plot them correctly.

Usage

```
turnmat(x)
```

Arguments

`x`	Matrix to be turned.

Value

Matrix `x`, turned by 90 degrees counter-clockwise.

Examples

```
set.seed(987)
sampleMat <- matrix(stats::rnorm(100), nrow = 10)
sampleMatTurn <- turnmat(x = sampleMat)
```

Index