Package ‘MassSpecWavelet’

April 26, 2017

Type Package
Title Mass spectrum processing by wavelet-based algorithms
Version 1.42.0
Date 2014-03-18
Depends waveslim
Suggests xcms, caTools
Author Pan Du, Warren Kibbe, Simon Lin
Maintainer Pan Du <dupan.mail@gmail.com>
Description Processing Mass Spectrometry spectrum by using wavelet based algorithm
License LGPL (>= 2)
biocViews MassSpectrometry, Proteomics
NeedsCompilation yes

R topics documented:

MassSpecWavelet-package ........................................ 2
cwt ................................................................. 2
de...
MassSpecWavelet-package

Peak detection of mass spectrum by Wavelet transform based methods

Description

Process Mass Spectrum (MS) by Wavelet Transforms-based algorithms

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>MassSpecWavelet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>1.0.4</td>
</tr>
<tr>
<td>Date:</td>
<td>2007-04-05</td>
</tr>
<tr>
<td>License:</td>
<td>GPL 2 or newer</td>
</tr>
</tbody>
</table>

MassSpecWavelet R package is aimed to process Mass Spectrometry (MS) data mainly based on Wavelet Transforms. The current version only supports the peak detection based on Continuous Wavelet Transform (CWT). Future versions will include more functions covering entire MS data processes.

Author(s)

Pan Du, Simon Lin
Maintainer: Pan Du <dupan@northwestern.edu>

References


Examples

data(exampleMS)
SNR.Th <- 3
peakInfo <- peakDetectionCWT(exampleMS, SNR.Th=SNR.Th)
majorPeakInfo = peakInfo$majorPeakInfo
peakIndex <- majorPeakInfo$peakIndex
plotPeak(exampleMS, peakIndex, main=paste('Identified peaks with SNR >', SNR.Th))

CWT

Continuous Wavelet Transform (CWT)

Description

CWT(Continuous Wavelet Transform) with Mexican Hat wavelet (by default) to match the peaks in Mass Spectrometry spectrum
Usage

cwt(ms, scales = 1, wavelet = "mexh")

Arguments

ms Mass Spectrometry spectrum (a vector of MS intensities)
scales a vector represents the scales at which to perform CWT.
wavelet The wavelet base, Mexican Hat by default. User can provide wavelet \( \Psi(x) \) as a form of two row matrix. The first row is the \( x \) value, and the second row is \( \Psi(x) \) corresponding to \( x \).

Value

The return is the 2-D CWT coefficient matrix, with column names as the scale. Each column is the CWT coefficients at that scale.

Author(s)

Pan Du, Simon Lin

Examples

data(exampleMS)
scales <- seq(1, 64, 3)
wCoefs <- cwt(exampleMS[5000:11000], scales=scales, wavelet='mexh')

## Plot the 2-D CWT coefficients as image (It may take a while!)
xTickInterval <- 1000
image(5000:11000, scales, wCoefs, col=terrain.colors(256), axes=FALSE, xlab='m/z index', ylab='CWT coefficient scale', main='CWT coefficients')
axis(1, at=seq(5000, 11000, by=xTickInterval))
axis(2, at=c(1, seq(10, 64, by=10)))
box()

exampleMS

An example mass spectrum

Description

An example mass spectrum from CAMDA 2006. All-in-1 Protein Standard II (Ciphergen Cat. \# C100-0007) were measured on Ciphergen NP20 chips. There are 7 polypeptides in the sample with m/z values of 7034, 12230, 16951, 29023, 46671, 66433, 147300.

Usage

data(exampleMS)

Format

A numeric vector represents the mass spectrum with equal sample intervals.

Source

**extendLength**  
*Extend the length of a signal or matrix*

**Description**  
Extend the length of a signal or matrix by row

**Usage**  
`extendLength(x, addLength = NULL, method = c("reflection", "open", "circular"), direction = c("right", "left", "both"))`

**Arguments**  
- `x`: a vector or matrix with column with each column as a signal
- `addLength`: the length to be extended
- `method`: three methods available, `c("reflection", "open", "circular")`. By default, it is "reflection".
- `direction`: three options available: `c("right", "left", "both")`

**Value**  
return the extended vector or matrix.

**Author(s)**  
Pan Du

**See Also**  
`extendNBase`

**Examples**  
```r  
# a = matrix(rnorm(9), 3)  
# extendLength(a, 3, direction='right')  ## not exposed function
```

---

**extendNBase**  
*Extend the row number of a matrix as the exponential of base N*

**Description**  
Extend the data as the exponential of base N by increasing row number.

**Usage**  
`extendNBase(x, nLevel=1, base=2, ...)`
getLocalMaximumCWT

Arguments

- `x` - data matrix
- `nLevel` - the level of DWT decomposition. Basically, it is equivalent to changing the 'base' as base\(^nLevel
- `base` - the base, 2 by default
- ... - other parameters of used by `extendLength`

Details

The method 'open' is padding the the matrix with the last row.

Value

Return a extended matrix

Author(s)

Pan Du

See Also

- `extendLength`

Examples

```r
# a = matrix(rnorm(9), 3)
# extendNBase(a) ## not exposed function
```

getLocalMaximumCWT

Identify the local maximum of each column in 2-D CWT coefficients matrix

Description

Identify the local maximum of each column in 2-D CWT coefficients matrix by using a slide window. The size of slide window linearly changes from the coarse scale (bigger window size) to detail scale. The scale of CWT increases with the column index.

Usage

`getLocalMaximumCWT(wCoefs, minWinSize= 5, amp.Th = 0)`

Arguments

- `wCoefs` - 2-D CWT coefficients, each column corresponding to CWT coefficient at one scale. The column name is the scale.
- `minWinSize` - The minimum slide window size used.
- `amp.Th` - The minimum peak amplitude.
getRidge

Identify ridges based on the local maximum matrix

Description
Identify ridges by connecting the local maximum of 2-D CWT coefficients from the coarse scale to detail scale. The local maximum matrix is returned from `getLocalMaximumCWT`.

Usage
```r
getRidge(localMax, iInit = ncol(localMax), step = -1, iFinal = 1, minWinSize= 5, gapTh = 3, skip = NULL)
```

Arguments
- `localMax`: The local maximum matrix is returned from `getLocalMaximumCWT` with 1 represents maximum, others are 0.
- `iInit`: The start column to search ridge. By default, it starts from the coarsest scale level.
- `step`: Search step. -1 by default, which means searching from coarse scale to detail scale column by column.
- `iFinal`: The final column index of search ridge.
- `minWinSize`: The minimum slide window size used.
- `gapTh`: The gap allowed during searching for ridge. 3 by default.
- `skip`: The column to be skipped during search.

Value
Return a list of ridge. As some ridges may end at the scale larger than 1, in order to keep the uniqueness of the ridge names, we combined the smallest scale of the ridge and m/z index of the peak at that scale together to name the ridges. For example the ridge name "1\_653" means the peak ridge ends at the CWT scale 1 with m/z index 653 at scale 1.
getRidgeLength

Author(s)
Pan Du, Simon Lin

References

See Also
getLocalMaximumCWT, identifyMajorPeaks

Examples

```r
data(exampleMS)
scales <- seq(1, 64, 3)
wCoefs <- cwt(exampleMS[5000:11000], scales=scales, wavelet='mexh')

localMax <- getLocalMaximumCWT(wCoefs)
ridgeList <- getRidge(localMax)
plotRidgeList(ridgeList)
```

getRidgeLength

Estimate the length of the ridge

Description
Estimate the length of the ridge line, which is composed of local maxima at adjacent CWT scales. The ridge line is cut off at the end point, whose amplitude divided by the maximum ridge amplitude is larger than the cutoff amplitude ratio threshold (0.5 by default).

Usage

```r
getRidgeLength(ridgeList, Th = 0.5)
```

Arguments

- `ridgeList`: a list of identified ridges
- `Th`: the cutoff amplitude ratio (the amplitude divided by the maximum amplitude of the ridge) threshold of the ridge line end.

Value

a vector of estimated ridge length

Author(s)
Pan Du
identifyMajorPeaks

Identify peaks based on the ridges in 2-D CWT coefficient matrix

Description
Identify the peaks based on the ridge list (returned by getRidge) in 2-D CWT coefficient matrix and estimated Signal to Noise Ratio (SNR)

Usage
identifyMajorPeaks(ms, ridgeList, wCoefs, scales = as.numeric(colnames(wCoefs)), SNR.Th = 3, peakScaleRange = 32, nearbyPeak = FALSE, nearbyWinSize = 100, winSize.noise = 500, SNR.method = "quantile", minNoiseLevel = 0.001)

Arguments
- ms: the mass spectrometry spectrum
- ridgeList: returned by getRidge
- wCoefs: 2-D CWT coefficients
- scales: scales of CWT, by default it is the colnames of wCoefs
- SNR.Th: threshold of SNR
- peakScaleRange: the CWT scale range of the peak.
- nearbyPeak: whether to identify nearby peaks
- nearbyWinSize: window size for nearby peak identification
- winSize.noise: window size for noise estimation
- SNR.method: method for SNR estimation
- minNoiseLevel: minimum noise level

getRidgeValue

Get the CWT coefficient values corresponding to the peak ridge

Description
Get the CWT coefficient values corresponding to the peak ridge

Usage
getRidgeValue(ridgeList, wCoefs, skip = 0)

Arguments
- ridgeList: a list of ridge lines
- wCoefs: 2-D CWT coefficients
- skip: the CWT scale level to be skipped, by default the 0 scale level (raw spectrum) is skipped.

Value
A list of ridge values corresponding to the input ridgeList.

Author(s)
Pan Du
**identifyMajorPeaks**

- **nearbyPeak**: determine whether to include the small peaks close to large major peaks.
- **nearbyWinSize**: the window size to determine the nearby peaks. Only effective when nearbyPeak is true.
- **winSize.noise**: the local window size to estimate the noise level.
- **SNR.method**: method to estimate noise level. Currently, only 95 percentage quantile is supported.
- **minNoiseLevel**: the minimum noise level used in calculating SNR, i.e., if the estimated noise level is less than "minNoiseLevel", it will use "minNoiseLevel" instead. If the noise level is less than 0.5, it will be treated as the ratio to the maximum amplitude of the spectrum.

**Details**

The determination of the peaks is based on three rules: Rule 1: The maximum ridge scale of the peak should larger than a certain threshold Rule 2: Based on the scale of the peak (corresponding to the maximum value of the peak ridge) should be within certain range Rule 3: Based on the peak SNR

**Value**

Return a list with following elements:

- **peakIndex**: the m/z indexes of the identified peaks.
- **peakCenterIndex**: the m/z indexes of peak centers, which correspond to the maximum on the ridge. peakCenterIndex includes all the peaks, not just the identified major peaks.
- **peakCenterValue**: the CWT coefficients (the maximum on the ridge) corresponding to peakCenterIndex.
- **peakSNR**: the SNR of the peak, which is the ratio of peakCenterValue and noise level.
- **peakScale**: the estimated scale of the peak, which corresponds to the peakCenterIndex.
- **potentialPeakIndex**: the m/z indexes of all potential peaks, which satisfy all requirements of a peak without considering its SNR. Useful, if you want to change to a lower SNR threshold later.
- **allPeakIndex**: the m/z indexes of all the peaks, whose order is the same as peakCenterIndex, peakCenterValue, peakSNR and peakScale.

All of these return elements have peak names, which are the same as the corresponding peak ridges. see getRidge for details.

**Author(s)**

Pan Du, Simon Lin

**References**

localMaximum

Identify local maximum within a slide window.

Description
Find local maximum by transform the vector as matrix, then get the the maximum of each column. This operation is performed twice with vector shifted half of the winSize.

Usage
localMaximum(x, winSize = 5)

Arguments
x a vector represents a signal profile
winSize the slide window size, 5 by default.

Details
Instead of find the local maximum by a slide window, which slide all possible positions, we find local maximum by transform the vector as matrix, then get the the maximum of each column. This operation is performed twice with vector shifted half of the winSize. The main purpose of this is to increase the efficiency of the algorithm.

Value
Return a vector with the same length of the input x. The position of local maximum is set as 1, 0 else where.

Author(s)
Pan Du
mzInd2vRange

See Also
gLocalMaximumCWT

Examples

```r
x <- rnorm(200)
lmax <- localMaximum(x, 5)
maxInd <- which(lmax > 0)
plot(x, type='l')
points(maxInd, x[maxInd], col='red')
```

Description

Match m/z index to m/z value with a certain error range

Usage

```r
mzInd2vRange(mzInd, error = 0.003)
```

Arguments

- `mzInd`: a vector of m/z index
- `error`: error range

Value

return a vector of sorted m/z values

Author(s)

Pan Du

See Also

mzV2indRange
mzV2indRange  \hspace{1cm} \textit{Match m/z value to m/z index with a certain error range}

**Description**

Match m/z value to m/z index with a certain error range

**Usage**

```r
mzV2indRange(mzV, error = 0.003)
```

**Arguments**

- `mzV`: a vector of m/z value
- `error`: error range

**Value**

return a vector of sorted m/z indexes

**Author(s)**

Pan Du

**See Also**

- `mzInd2vRange`

---

**peakDetectionCWT**  \hspace{1cm} \textit{The main function of peak detection by CWT based pattern matching}

**Description**

This function is a wrapper of `cwt`, `getLocalMaximumCWT`, `getRidge`, `identifyMajorPeaks`

**Usage**

```r
peakDetectionCWT(ms, scales = c(1, seq(2, 30, 2), seq(32, 64, 4)), SNR.Th = 3, nearbyPeak = TRUE, peakScaleRange = 5, amp.Th = 0.01, minNoiseLevel = amp.Th/SNR.Th, ridgeLength = 24, peakThr=NULL, tuneIn = FALSE, ...)
```

**Arguments**

- `ms`: the mass spectrometry spectrum
- `scales`: scales of CWT
- `SNR.Th`: SNR (Signal to Noise Ratio) threshold
- `nearbyPeak`: Determine whether to include the nearby small peaks of major peaks. TRUE by default
- `peakScaleRange`: the scale range of the peak. larger than 5 by default.
- `amp.Th`: the minimum required relative amplitude of the peak (ratio to the maximum of CWT coefficients)
peakDetectionCWT

minNoiseLevel: the minimum noise level used in computing the SNR.

ridgeLength: the minimum highest scale of the peak in 2-D CWT coefficient matrix.

peakThr: Minimal absolute intensity (above the baseline) of peaks to be picked. If this value is provided, then the smoothing function `sav.gol` will be called to estimate the local intensity. (Added based on the suggestion and code of Steffen Neumann).

tuneIn: determine whether to tune in the parameter estimation of the detected peaks.

Value:

- `majorPeakInfo`: return of `identifyMajorPeaks`
- `ridgeList`: return of `getRidge`
- `localMax`: return of `getLocalMaximumCWT`
- `wCoefs`: 2-D CWT coefficient matrix, see `cwt` for details.

Author(s):

Pan Du, Simon Lin

References:


See Also:

- `cwt`, `getLocalMaximumCWT`, `getRidge`, `identifyMajorPeaks`

Examples:

```r
data(exampleMS)
SNR.Th <- 3
peakInfo <- peakDetectionCWT(exampleMS, SNR.Th=SNR.Th)
majorPeakInfo = peakInfo$majorPeakInfo
peakIndex <- majorPeakInfo$peakIndex
plotPeak(exampleMS, peakIndex, main=paste(quote(Identified peaks with SNR >', SNR.Th)))

## In some cases, users may want to add peak filtering based on the absolute peak amplitude
peakInfo <- peakDetectionCWT(exampleMS, SNR.Th=SNR.Th, peakThr=500)
majorPeakInfo = peakInfo$majorPeakInfo
peakIndex <- majorPeakInfo$peakIndex
plotPeak(exampleMS, peakIndex, main=paste(quote(Identified peaks with SNR >', SNR.Th)))
```
plotLocalMax

Plot the local maximum matrix

Description

Plot the local maximum matrix of 2-D CWT coefficients returned by getLocalMaximumCWT

Usage

plotLocalMax(localMax, wCoefs = NULL, range = c(1, nrow(localMax)), colorMap = "RYB", main = NULL, cex = 3, pch = ".", ...)

Arguments

localMax local maximum matrix of 2-D CWT coefficients returned by getLocalMaximumCWT
wCoefs 2-D CWT coefficients
range plot range of m/z index
colorMap the colormap used in plotting the points
main parameter of plot
cex parameter of plot
pch parameter of plot
... other parameters of points

Author(s)

Pan Du

See Also

getLocalMaximumCWT

Examples

data(exampleMS)
scales <- seq(1, 64, 3)
wCoefs <- cwt(exampleMS[5000:11000], scales=scales, wavelet='mexh')

localMax <- getLocalMaximumCWT(wCoefs)
plotLocalMax(localMax)
plotPeak

Plot the identified peaks over the spectrum

Description

Plot the identified peaks over the spectrum. The identified peaks are returned by `peakDetectionCWT` or `identifyMajorPeaks`.

Usage

```r
plotPeak(ms, peakIndex = NULL, mz = 1:length(ms), range = c(min(mz), max(mz)), method = c("p", "l"), main = NULL, log = "", ...)```

Arguments

- `ms`: the MS spectrum
- `peakIndex`: m/z indexes of the identified peaks
- `mz`: m/z value correspond to m/z index
- `range`: the plot range of m/z value
- `method`: plot method of the identified peaks. method 'p' plot circles on the peaks; method 'l' add vertical lines over the peaks.
- `main`: parameter of `plot`
- `log`: parameter of `plot`
- `...`: other parameters of `points`

Author(s)

Pan Du

See Also

`peakDetectionCWT`, `identifyMajorPeaks`

Examples

```r
data(exampleMS)
SNR.Th <- 3
peakInfo <- peakDetectionCWT(exampleMS, SNR.Th=SNR.Th)
majorPeakInfo = peakInfo$majorPeakInfo
peakIndex <- majorPeakInfo$peakIndex
plotPeak(exampleMS, peakIndex, main=paste('Identified peaks with SNR >', SNR.Th))
```
plotRidgeList

Plot the ridge list

Description

Plot the ridge list returned by getRidge

Usage

plotRidgeList(ridgeList, wCoefs = NULL, range = NULL, colorMap = "RYB", main = NULL, pch = ".", cex = 2, ...)

Arguments

ridgeList returned by getRidge
wCoefs 2-D CWT coefficients
range plot range of m/z index
colorMap colorMap to plot the points of local maximum
main parameter of plot
pch parameter of plot
cex parameter of plot
... other parameters of points

Author(s)

Pan Du

See Also

getRidge

Examples

data(exampleMS)
scales <- seq(1, 64, 3)
wCoefs <- cwt(exampleMS[5000:11000], scales=scales, wavelet='mexh')

localMax <- getLocalMaximumCWT(wCoefs)
ridgeList <- getRidge(localMax)
plotRidgeList(ridgeList)
**sav.gol**

*Estimate the baseline by using Savitzky-Golay Algorithm*

**Description**

Estimate the baseline by using Savitzky-Golay Algorithm

**Usage**

```
sav.gol(T, fl, forder = 4, dorder = 0)
```

**Arguments**

- `T` vector of signals to be filtered
- `fl` filter length (for instance fl = 51..151)
- `forder` filter order (2 = quadratic filter, 4 = quartic)
- `dorder` derivative order (0 = smoothing, 1 = first derivative, etc.)

**Value**

The return is a smoothed vector (baseline).

**Note**

This function was added by Steffen Neumann. We appreciated his help to make the package better.

**Author(s)**

Steffen Neumann <sneumann@ipb-halle.de>

---

**smoothDWT**

*smooth (denoise) the spectrum by DWT (Discrete Wavelet Transform)*

**Description**

Smooth (denoise) the spectrum by DWT (Discrete Wavelet Transform)

**Usage**

```
smoothDWT(ms, nLevel = 6, wf = "la8", localNoiseTh = seq(1, 0, by = -0.2), localWinSize = 500, globalWinSize = 500)
```
tuneInPeakInfo

Arguments

ms  a vector representing the mass spectrum
nLevel  the level of DWT decomposition
wf  the name of wavelet for DWT
localNoiseTh  local noise level threshold
localWinSize  local window size for estimate local noise threshold
globalNoiseTh  global noise level threshold
smoothMethod  the method used for denoising. 'hard' means keeping the dwt coefficients higher than the threshold unchanged; 'soft' means the dwt coefficients higher than the threshold were subtracted by the threshold.
method  'dwt' or 'modwt' used for decomposition

Value

return the smoothed mass spectrum with the 'detail' component of DWT as an attribute 'detail'.

Author(s)

Pan Du

detail

Tune in the peak information: peak position and peak scale

Description

Based on the identified peak position, more precise estimation of the peak information, i.e., peak position and peak scale, can be got by this function. The basic idea is to cut the segment of spectrum near the identified peaks, and then do similar procedures as peakDetectionCWT, but with more detailed scales around the estimated peak scale.

Usage

tuneInPeakInfo(ms, majorPeakInfo = NULL, peakIndex = NULL, peakScale = NULL, maxScale = 128, ...)

Arguments

ms  the mass spectrometry spectrum
majorPeakInfo  return of identifyMajorPeaks
peakIndex  the m/z index of the identified peaks
peakScale  the scales of the identified peaks
maxScale  the maximum scale allowed for the peak
...  other parameters of used by getLocalMaximumCWT, getRidge, identifyMajorPeaks

Details

The majorPeakInfo or peakIndex and peakScale must be provided.
tuneInPeakInfo

Value

peakCenterIndex the updated peak center m/z index
peakScale the updated peak scale
peakValue the corresponding peak value

Author(s)

Pan Du

References


See Also

peakDetectionCWT

Examples

data(exampleMS)
SNR.Th <- 3
peakInfo <- peakDetectionCWT(exampleMS, SNR.Th=SNR.Th)
majorPeakInfo <- peakInfo$majorPeakInfo
betterPeakInfo <- tuneInPeakInfo(exampleMS, majorPeakInfo)
plot(500:length(exampleMS), exampleMS[500:length(exampleMS)], type='l', log='x')
abline(v=betterPeakInfo$peakCenterIndex, col='red')
Index

*Topic **datasets**
   exampleMS, 3

*Topic **hplot**
   plotLocalMax, 14
   plotPeak, 15
   plotRidgeList, 16

*Topic **methods**
   cwt, 2
   extendLength, 4
   extendNBase, 4
   getLocalMaximumCWT, 5
   getRidge, 6
   getRidgeLength, 7
   getRidgeValue, 8
   identifyMajorPeaks, 8
   localMaximum, 10
   mzInd2vRange, 11
   mzV2indRange, 12
   peakDetectionCWT, 12
   sav.gol, 13, 17
   smoothDWT, 17
   tuneInPeakInfo, 10, 18

*Topic **package**
   MassSpecWavelet-package, 2

   cwt, 2, 12, 13
   exampleMS, 3
   extendLength, 4, 5
   extendNBase, 4, 4
   getLocalMaximumCWT, 5, 6, 7, 11–14, 18
   getRidge, 6, 8, 9, 12, 13, 16, 18
   getRidgeLength, 7
   getRidgeValue, 8
   identifyMajorPeaks, 7, 8, 12, 13, 15, 18
   localMaximum, 6, 10
   MassSpecWavelet
      (MassSpecWavelet-package), 2
   MassSpecWavelet-package, 2
   mzInd2vRange, 11, 12
   mzV2indRange, 11, 12
   peakDetectionCWT, 10, 12, 15, 18, 19
   plot, 14–16
   plotLocalMax, 14
   plotPeak, 15
   plotRidgeList, 16
   points, 14–16
   sav.gol, 13, 17
   smoothDWT, 17
   tuneInPeakInfo, 10, 18