

Package ‘spectralPhasoR’

October 23, 2018

Type Package

Title spectral Phasor representation of sPAINT data

Version 0.1.1

Author JuG

Maintainer Julien Godet <julien.godet@unistra.fr>

Description Compute and analyse spectrally resolved single molecule localisation microscopy (SR-SMLM) data using spectral phasor.

License GPL3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.0

R topics documented:

calcAngle	2
calcModule	2
calcPhasor	3
calibration	4
checkLocOverlap	4
checkLocOverlapInFrame	5
checkLocPosition	5
clusterBeads	6
getCorrectedLambda	6
getLoc	7
getLocalMax	7
getLocProfile	8
lambda2rgb	8
localMaxima	9
plotPhasor	9
plotSpectra	10
plotSpectraLambda	10
setPhasor	11
Index	12

calcAngle*Compute angle value in degree from phasor coordinates*

Description

Compute angle value in degree from phasor coordinates

Usage

```
calcAngle(gPhasor, sPhasor, rad = FALSE)
```

Arguments

gPhasor	gPhasor
sPhasor	sPhasor

Details

Use the modulo to get angle

Author(s)

JuG

Examples

```
all.equal(calcAngle(0,1, rad=T), pi/2)
calcAngle(1,1)
calcAngle(c(-1,2),-1)
```

calcModule*Compute module value from phasor coordinates*

Description

Compute module value from phasor coordinates

Usage

```
calcModule(gPhasor, sPhasor)
```

Arguments

gPhasor	gPhasor
sPhasor	sPhasor

Author(s)

JuG

Examples

```
calcModule(1,1)
all.equal(calcModule(1,-1), sqrt(2))
```

`calcPhasor`*Compute spectral phasor coordinates*

Description

The spectral phasor transformation calculates the Fourier sine and cosine transforms of a given spectrum. For each Fourier harmonic, two coordinates are calculated: g corresponds to the cosine transform (x coordinate in a polar plot) and s corresponds to the sine transform (y coordinate in a polar plot)

Usage

```
calcPhasor(Int, lambda, n = 1, col, point = FALSE)
```

Arguments

Int	fluorescence intensities for each lambda value
lambda	wavelengths
n	= harmonic

Value

matrix

Author(s)

JuG

Examples

```
Int <- c(4,6,5,8,10,20,35,50,60,53,45,32,28,30,25,22,17,10,5,4,3,4,3,2)
lambda <- seq(580,680,length=24)
calcPhasor(Int,lambda)
```

calibration	<i>Do something</i>
-------------	---------------------

Description

Do something

Usage

```
calibration(localisation, image, profileLength, deltaSpectre,  
  lambdaRange = list(range1 = c(240, 255), range2 = c(270, 285), range3 =  
    c(315, 335)), lref = c(512, 581.5, 676.5), showPlot = TRUE, delta)
```

Arguments

showPlot	if true plot local maximum (in px) as a function of lambda ref
----------	--

Author(s)

JuG

checkLocOverlap	<i>Check for spectral overlap between localisations</i>
-----------------	---

Description

Check for spectral overlap between localisations

Usage

```
checkLocOverlap(localisation, profileLength, clean = TRUE)
```

Author(s)

JuG

`checkLocOverlapInFrame`*Check for spectral overlap between localisation in a frame*

Description

Check for spectral overlap between localisation in a frame

Usage

```
checkLocOverlapInFrame(localisation, profileLength)
```

Arguments

`localisation` localisation data.frame of a given frame

Value

1/0 (overlap/non-overlap)

Author(s)

JuG

`checkLocPosition`*Return a data.frame without the localisations with incomplete profile*

Description

Return a data.frame without the localisations with incomplete profile

Usage

```
checkLocPosition(localisation, deltaSpectre, profileLength)
```

Arguments

`localisation` localisation data.frame with clusters

Author(s)

JuG

clusterBeads	<i>Cluster localisations using DBSCAN</i>
--------------	---

Description

Cluster localisations using DBSCAN

Usage

```
clusterBeads(loc)
```

Arguments

loc	a localisation data.frame
-----	---------------------------

Value

void

Author(s)

JuG

getCorrectedLambda	<i>Recalculate the wavelength values for a given ZOZ1 distance range</i>
--------------------	--

Description

Recalculate the wavelength values for a given ZOZ1 distance range

Usage

```
getCorrectedLambda(pixPosition = 240:340, X = 101, Y = 302,  
  modC = modCalib)
```

Arguments

X	X position
Y	Y position
modC	calibration model

Author(s)

JuG

Examples

```
getCorrectedLambda(pixPosition = 285+c(-60:60), X = 120, Y = 302, modC = modCalib)  
getCorrectedLambda(pixPosition = 240:340, X = 1, Y = 2, modC = modCalib)
```

`getLoc`*Import localisations from a PeakFit .xls output file*

Description

Import localisations from a PeakFit .xls output file

Usage

```
getLoc(path)
```

Arguments

path path to a PeakFit .xls output file

Value

data.frame

Author(s)

JuG

Examples

```
path <- "/Users/jgodet/Documents/1488/TSBref_1/TSBref_1_MMStack_Pos0.ome.tif.results.xls"
loc <- getLoc(path)
```

`getLocalMax`*Do something*

Description

Do something

Usage

```
getLocalMax(sProfile, range, subpx.resol = 0.1, lineProfile = FALSE,
  drawLocMax = FALSE)
```

Author(s)

JuG

Examples

```
y = getLocProfile(localisation = locRef[2,], image = stackImRef, profileLength = 50, deltaSpectre=280)
getLocalMax(y, lineProfile = TRUE)
getLocalMax(y, range=c(240,250), lineProfile = TRUE)
getLocalMax(y, range=c(270,285), lineProfile = TRUE)
getLocalMax(y, range=c(310,335), lineProfile = TRUE)
```

getLocProfile	<i>Extract the spectral decomposition profile of a given localisation</i>
---------------	---

Description

Extract the spectral decomposition profile of a given localisation

Usage

```
getLocProfile(localisation, image = stackIm, deltaSpectre, profileLength,
              delta)
```

Arguments

localisation	a localisation (at least define by Frame, X and Y coordinates)
image	image Stack
deltaSpectre	shift in pixels between Z0 and Z1
profileLength	length of the profile in px
delta	additional Y shift (account for misaligned Z0 and Z1)

Value

data.frame

Author(s)

JuG

Examples

```
deltaSpectre = 280
profileLength = 50
getLocProfile(localisation = locRef[1,], image = stackImRef, deltaSpectre = deltaSpectre, profileLength = profileLength)
```

lambda2rgb	<i>Transform wavelength to real color code in rgb</i>
------------	---

Description

Transform wavelength to real color code in rgb

Usage

```
lambda2rgb(lambda, alpha)
```

Arguments

lambda	wavelength (in nm)
alpha	transparency

Author(s)

JuG

Examples

```
lambda2rgb(lambda = 488,alpha = 1)
lambda2rgb(lambda = 532,alpha = 1)
```

localMaxima*Find local maxima of a vector sequence*

Description

grabbed somewhere on the web

Usage

```
localMaxima(x)
```

Arguments

x a vector of value for which local maxima are searched

Author(s)

JuG

plotPhasor*Plot phasor*

Description

Plot phasor

Usage

```
plotPhasor(gPhasor, sPhasor, ...)
```

Arguments

gPhasor gPhasor corrdinates
sPhasor sPhasor corrdinates

Value

a phasor graph

Author(s)

JuG

plotSpectra	<i>Plot spectra for an index of localisations</i>
-------------	---

Description

Plot spectra for an index of localisations

Usage

```
plotSpectra(locData, stackIm, index, profileLength = 50,
  deltaSpectre = 280, delta, modCalib = modCalib, wavelength = TRUE,
  ...)
```

Author(s)

JuG

Examples

```
plotSpectra(locData = locRef, stackIm = stackImRef, profileLength = 60, deltaSpectre = 290, ylim=c(0.04,0.12))
plotSpectra(locData = locRef, stackIm = stackImRef, index=c(12,13), profileLength = 30, deltaSpectre = 280)
index <- which(locRef$beads==2)
plotSpectra(locData = locRef, stackIm = stackImRef, index=index, profileLength = 20, deltaSpectre = 275, ylim=c(0
```

plotSpectraLambda	<i>Do something</i>
-------------------	---------------------

Description

Do something

Usage

```
plotSpectraLambda(locData, stackIm, index, profileLength = 50,
  deltaSpectre = 280, ylim, delta, lambda, norm = TRUE,
  coul = "black", add = FALSE, xlim, smoothY = FALSE, cex.lab = 1)
```

Author(s)

JuG

setPhasor	<i>Add phasor coordinates and an estimation of the fluorescence emission maximum to the localisation data.frame</i>
-----------	---

Description

Add phasor coordinates and an estimation of the fluorescence emission maximum to the localisation data.frame

Usage

```
setPhasor(loc2, stackIm, deltaSpectre, profileLength, modCalib, maxRange,  
          normalizeVal = NULL, nphasor = 1, delta)
```

Arguments

stackIm	image stack
deltaSpectre	ZOZ1 distance (estimation)
profileLength	half profile length
modCalib	calibration model
maxRange	range value of the expected emission maximum
loc	localisation data.frame

Value

data.frame with localisation parameters + deltaUsed + Phasor coordinates (g and s) + Maximum fluorescence emission wavelength (in pixels MaxEmPix or in nanometers MaxEm.nm) and the intensity-weighted averaged emission wavelength

Author(s)

JuG

Index

calcAngle, [2](#)
calcModule, [2](#)
calcPhasor, [3](#)
calibration, [4](#)
checkLocOverlap, [4](#)
checkLocOverlapInFrame, [5](#)
checkLocPosition, [5](#)
clusterBeads, [6](#)

getCorrectedLambda, [6](#)
getLoc, [7](#)
getLocalMax, [7](#)
getLocProfile, [8](#)

lambda2rgb, [8](#)
localMaxima, [9](#)

plotPhasor, [9](#)
plotSpectra, [10](#)
plotSpectraLambda, [10](#)

setPhasor, [11](#)