

# Package ‘FISHalyseR’

May 24, 2024

**Type** Package

**Title** FISHalyseR a package for automated FISH quantification

**Version** 1.39.0

**Date** 2015-04-08

**Author** Karesh Arunakirinathan <akaresh88@gmail.com>, Andreas Heindl  
<andreas.heindl@icr.ac.uk>

**Maintainer** Karesh Arunakirinathan <akaresh88@gmail.com>, Andreas  
Heindl <andreas.heindl@icr.ac.uk>

**Description** FISHalyseR provides functionality to process and analyse  
digital cell culture images, in particular to quantify FISH  
probes within nuclei. Furthermore, it extract the spatial  
location of each nucleus as well as each probe enabling spatial  
co-localisation analysis.

**VignetteBuilder** knitr

**License** Artistic-2.0

**Depends** EBImage,abind

**Suggests** knitr

**biocViews** CellBiology

**git\_url** <https://git.bioconductor.org/packages/FISHalyseR>

**git\_branch** devel

**git\_last\_commit** f17941c

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.20

**Date/Publication** 2024-05-24

## Contents

analyseParticles . . . . .	2
calculateMaxEntropy . . . . .	3
calculateThreshold . . . . .	4
computeIlluminationCorrection . . . . .	5
processFISH . . . . .	5

---

analyseParticles	<i>Analyse</i>
------------------	----------------

---

### Description

Cleans a given binary image according to area criteria specified by the user.

### Usage

```
analyseParticles(Image,MaxSize,MinSize, isMask)
```

### Arguments

Image	Binary image
MaxSize	Maximum area allowed for objects
MinSize	Minimum area allowed for objects
isMask	In case isMask=1, the function assumes that the binary images contains nuclei. Nuclei with an area smaller than MaxSize and greater than MinSize will be removed. If isMask=0, the function assumes that the binary images contains probes and subsequently probes with an area smaller than MinSize or larger than MaxSize are removed

### Value

Returns a labeled image

### Author(s)

Karesh Arunakirinathan

### Examples

```
f = system.file( "extdata", "SampleFISHgray.jpg", package="FISHalyseR")
img = readImage(f)

anaImg <- analyseParticles(img, 20000, 1000,0)
## anaImg contains now the cleaned-up image
```

---

calculateMaxEntropy     *Max Entropy thresholding*

---

**Description**

The function converts a grayscale image to a binary image by computing a threshold using the Max Entropy method.

**Usage**

```
calculateMaxEntropy(Image)
```

**Arguments**

Image            grayscale image

**Details**

Max Entropy thresholding can be used to detect the signals of probes in FISH cell culture images.

**Value**

The function returns the threshold value

**Author(s)**

Karesh Arunakirinathan

**References**

J.N KANPUR, P.K SHAOO, A.K.C WONG: A New Method for Gray-Level picture thresholding Using the Entropy of the Histogram. In COMPUTER VISION, GRAPHICS AND IMAGE PROCESSING,1985 p 273-285

**See Also**

calculateThreshold

**Examples**

```
f = system.file( "extdata", "SampleFISHgray.jpg", package="FISHalyseR")
img = readImage(f)

t = calculateMaxEntropy(img)

## Threshold grayscale image using the value computed by the Max Entropy method
img[img<t] <- 0
img[img>=t] <- 1
```

---

calculateThreshold     *Compute threshold using Otsu's method*

---

**Description**

Computes the binary image of a grayscale image by using Otsu thresholding

**Usage**

```
calculateThreshold(Image)
```

**Arguments**

Image            grayscale image

**Details**

The function computes a binary image using Otsu's method.

**Value**

calculateThreshold returns the threshold value

**Author(s)**

Karesh Arunakirinathan

**References**

Nobuyuki Otsu: A threshold selection method from grey level histograms. In: IEEE Transactions on Systems, Man, and Cybernetics. New York 9.1979, S.62-66. ISSN 1083-4419

**See Also**

calculateMaxEntropy

**Examples**

```
f = system.file( "extdata", "SampleFISHgray.jpg", package="FISHalyseR")
img = readImage(f)

t = calculateThreshold(img)

##Threshold image using the value computed via Otsu's method
img[img<t] <- 0
img[img>=t] <- 1
```

---

computeIlluminationCorrection  
*Multidimensional Illumination Correction*

---

**Description**

Function to compute the multidimensional illumination correction (MDIC) using a stack of images

**Usage**

```
computeIlluminationCorrection(Images,pattern='*',AmountOfFiles=6)
```

**Arguments**

Images	Directory containing the images
pattern	Filenames have to match the pattern specified here
AmountOfFiles	Limit the amount of files used to compute the illumination gradient

**Value**

```
computeIlluminationCorrection  
return the image containing the illumination background
```

**Author(s)**

Andreas Heindl

**Examples**

```
illuCorrection = dirname(system.file("extdata", "SampleFISHillu.jpg", package="FISHalyseR"))
```

---

processFISH                    *FISHalyseR - Automated fluorescence in situ hybridisation quantification in R*

---

**Description**

Function to automatically quantify FISH probes in cell-culture images.

**Usage**

```
processFISH(combinedImg, writedir, bgCorrMethod = list(1, 100), channelSignals = NULL,  
channelColours = NULL, sizeNucleus = c(5, 15000), sizeProbe = c(5, 100),  
gaussigma = 20, outputImageFormat = ".png")
```

**Arguments**

<code>combinedImg</code>	Composite image of all available channels
<code>writedir</code>	Target directory for output files
<code>bgCorrMethod</code>	Specifies the method used to correct for uneven background. Accepts only list types. Currently, four different methods are available: (1) Gaussian blurring, (2) Illumination correction image provided by the user, (3) multidimensional illumination correction (using a stack of images). In case no illumination correction should be applied, pass an empty list to the function
<code>channelSignals</code>	List of images containing the FISH probe
<code>channelColours</code>	List of colour vectors for each single channel
<code>sizeNucleus</code>	Minimum and maximum area (in pixel) of probes to be considered for further analysis
<code>sizeProbe</code>	Minimum and maximum area (in pixel) of probes to be considered for further analysis
<code>gaussigma</code>	Sigma of Gaussian used to blur the image
<code>outputImageFormat</code>	File format for the output image

**Value**

`processFISH` does not return any value

**Author(s)**

Karesh Arunakirinathan, Andreas Heindl

**See Also**

`computeIlluminationCorrection`, `analyseParticles`

**Examples**

```
## Specify illumination correction image
illuCorrection = system.file( "extdata", "SampleFISHillu.jpg", package="FISHalyseR")

## Composite image containing available channels
combinedImage <- system.file( "extdata", "SampleFISH.jpg", package="FISHalyseR")

## Single FISH channels containing the probe signals
red_Og <- system.file( "extdata", "SampleFISH_R.jpg", package="FISHalyseR")
green_Gn <- system.file( "extdata", "SampleFISH_G.jpg", package="FISHalyseR")

## Output directory
writedir = paste(tempdir(),sep='')

## Use provided illumination correction image
bgCorrMethod = list(2,illuCorrection)
```

```
## Colour vector for three different probe channels (red, green and blue)
channelColours = list(R=c(255,0,0),G=c(0,255,0))

## Add probe channels to list
channelSignals = list(red_0g,green_Gn)

## Minimum and maximum area allowed for nuclei respectively probes
sizecell = c(1000,20000)
sizeprobe= c(5,20)

## Call processFISH with the specified parameters
processFISH(combinedImage,writedir,bgCorrMethod,channelSignals,
            channelColours,sizecell,sizeprobe)
```

# Index

\* **misc, bwlabel**

analyseParticles, [2](#)

\* **misc**

calculateMaxEntropy, [3](#)

calculateThreshold, [4](#)

computeIlluminationCorrection, [5](#)

processFISH, [5](#)

analyseParticles, [2](#)

calculateMaxEntropy, [3](#)

calculateThreshold, [4](#)

computeIlluminationCorrection, [5](#)

processFISH, [5](#)