Package 'cyanoFilter'

January 9, 2020

Title Cyanobacteria Population Identification for Flow Cytometry

Version 0.1.3

Maintainer Oluwafemi Olusoji <oluwafemi.olusoji@uhasselt.be>

Description An approach to filter out and/or identify synechococcus type cyanobacteria cells from all particles measured via flow cytometry.
It combines known characteristics of these cyanobacteria strains alongside gating techniques developed by Mehrnoush, M. et al. (2015) <doi:10.1093/bioinformatics/btu677>
in the 'flowDensity' package to identify and separate these cyanobacteria cells from other cell types.
Aside the gating techniques in the 'flowDensity' package, an EM style clustering technique is also developed to identify these cyanobacteria cell populations.

URL https://github.com/fomotis/cyanoFilter

BugReports https://github.com/fomotis/cyanoFilter/issues

Depends R(>= 3.4), Biobase(>= 2.40.0)

Imports flowCore(>= 1.42.3), flowDensity (>= 1.10.0), graphics(>= 3.6.0), grDevices(>= 3.6.0), methods(>= 3.5.1), RColorBrewer(>= 1.1-2), Rdpack(>= 0.11-0), stats(>= 3.6.0), stringr(>= 1.3.1), utils(>= 3.6.0)

RdMacros Rdpack

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests dplyr, magrittr, knitr, rmarkdown, tidyr

VignetteBuilder knitr

NeedsCompilation no

Author Oluwafemi Olusoji [aut, cre],
Aerts Marc [ctb],
Delaender Frederik [ctb],
Neyens Thomas [ctb],
Spaak jurg [aut]

Repository CRAN
Date/Publication 2020-01-09 13:10:11 UTC

R topics documented:

`bs4_nc` *gates out or assign indicators to Synechococcus cyanobacteria cells in the bottom right of the 2-D space.*

Description

This function takes in a flowframe with debris removed and identifies Synechococcus cyanobacteria cell population in the provided frame.

Usage

```
bs4_nc(bs4bs5, p1, p2, others, to_retain = c("refined", "potential"))
```

Arguments

`bs4bs5`	flowframe with debris (left) removed.
`p1`	first flowcytometer channel that can be used to separate cells of interest from the rest, e.g. "RED.B.HLin".
`p2`	second flowcytometer channel that can be used to separate cells of interest from the rest, e.g. "YEL.B.HLin"
`others`	row numbers for non-debris events. This is provided by the `debris_nc` or `debris_inc` function.
`to_retain`	should potential candidates be retained or further gating be applied to filter out only certain Synechococcus cyanobacteria cells.

Details

The function uses the `getPeaks` and `deGate` functions in the *flowDensity* package to identify peaks and identify cut-off points between these peaks. This function is not designed to be called in isolation, if called in isolation an error will be returned. It is preferably called on the results from `debris_nc` or `debris_inc` function. A graph with horizontal and vertical lines used in separating the populations is returned and if *to_retain = "refined"*, a circle made of dashed lines is drawn around Synechococcus cyanobacteria cell population points.

Value

list containing;

- **syn_reduced -** flowframe containing only Synechococcus cyanobacteria cells
- **others_nk -** unidentified particle positions
- **syn_pos -** Synechococcus cyanobacteria cells positions
- **others_nk2 -** other unidentified particle positions

See Also

`bs5_nc`

---

bs5_nc *gates out or assign indicators to Synechococcus cyanobacteria cells in the top right of the 2-D space.*

Description

This function takes in a flowframe with debris removed and identifies ynechococcus cyanobacteria cell population in the provided frame.

Usage

```
bs5_nc(bs4bs5, p1, p2, others, to_retain = "potential")
```

Arguments

`bs4bs5`	flowframe with debris (left) removed.
`p1`	first flowcytometer channel that can be used to separate cells of interest from the rest, e.g. "RED.B.HLin".
`p2`	second flowcytometer channel that can be used to separate cells of interest from the rest, e.g. "YEL.B.HLin"
`others`	row numbers for non-debris events. This is provided by the debris_nc or debris_inc function.
`to_retain`	should potential candidates be retained or further gating be applied to filter out only certain Synechococcus cyanobacteria cells. @return list containing;

- **syn_reduced -** flowframe containing only BS5s
- **others_nk -** unidentified particle positions
- **syn_pos -** Synechococcus cyanobacteria cells positions
- **others_nk2 -** other unidentified particle positions

Details

The function uses the `getPeaks` and `deGate` functions in the *flowDensity* package to identify peaks and identify cut-off points between these peaks. This function is not designed to be called in isolation, if called in isolation an error will be returned. It is preferably called on the results from `debris_nc` or `debris_inc` function. A graph with horizontal and vertical lines used in separating the populations is returned and if *to_retain="refined"*, a circle made of dashed lines is drawn around Synechococcus cyanobacteria cell population points.

`celldebris_emclustering` *identifies Synechococcus cyanobacteria cells and Debris in a flowfile using an EM style algorithm.*

Description

separates BS4, BS5 and Debris population in a flowfile using an EM style algorithm. Algorithm starts with *ncluster* number of clusters and automatically reduces this number if need be.

Usage

```
celldebris_emclustering(flowfile, channels, mu = NULL, sigma = NULL,
  ncluster = 5, min.itera = 20, classifier = 0.8)
```

Arguments

`flowfile`	flowframe to be clustered.
`channels`	channels to use for the clustering
`mu`	pre-specified mean matrix for the clusters. Number of rows should equal ncluster and number of columns should equal length(channels). Defaults to NULL and will be computed from the data internally if left as NULL.
`sigma`	pre-specified list of variance-covariace matrix for the clusters. Each element of the list should contain a square matrix of variance-covariance matrix with length equal ncluster. Defaults to NULL and will be computed from the data internally if left as NULL.
`ncluster`	number of cluster desired.
`min.itera`	minimum number of EM iterations.
`classifier`	cells will be assigned to a cluster if belongs to that cluster by at least this probability. Only for plotting purposes.

Details

The function using EM algorithm involving mixtures of multivariate normals to separate the entire cell-population provided into cluster. The mvnorm function is used to compute the densities and only the probabilites of each point belonging to a cluster are returned as additional columns to the expression matrix of *result*.

Value

list containing;

- **percentages -** percentage of cells in each cluster
- **mus -** matrix of mean vectors for each cluster
- **sigmas -** list of variance-covariance matrix for each cluster
- **result -** flowframe with probabilities of each cluster added as columns to the expression matrix of the flowfile

See Also

celldebris_nc

Examples

```
flowfile_path <- system.file("extdata", "B4_18_1.fcs", package = "cyanoFilter",
              mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
                               dataset = 1) #FCS file contains only one data object
flowfile_nona <- cyanoFilter::nona(x = flowfile)
flowfile_noneg <- cyanoFilter::noneg(x = flowfile_nona)
flowfile_logtrans <- lnTrans(x = flowfile_noneg, c('SSC.W', 'TIME'))
cells_nonmargin <- cellmargin(flow.frame = flowfile_logtrans, Channel = 'SSC.W',
           type = 'estimate', y_toplot = "FSC.HLin")

emapproach <- celldebris_emclustering(flowfile = cells_nonmargin$reducedflowframe,
                    channels = c("RED.B.HLin", "YEL.B.HLin",
                    "FSC.HLin", "RED.R.HLin"),
                    ncluster = 5, min.itera = 20)
```

---

celldebris_nc *gates out or assign indicators to Synechococcus cyanobacteria cells.*

Description

This is a top-level function that calls other functions to identify cell population of interest.

Usage

```
celldebris_nc(flowframe, channel1 = "RED.B.HLin",
  channel2 = "YEL.B.HLin", interest = c("bottom-right", "top-right",
  "both-right"), to_retain = c("refined", "potential"))
```

Arguments

flowframe	flowframe with debris and Synechococcus cells.
channel1	first flowcytometer channel that can be used to separate cyanobacteria cells from the rest, e.g. "RED.B.HLin".
channel2	second flowcytometer channel that can be used to separate cyanobacteria cells from the rest, e.g. "YEL.B.HLin"
interest	a string indicating poistion of population of interest to be gated, can be "bottom-right", "top-right" or "both-right".
to_retain	should potential candidates be retained or further gating be applied to filter out only certain cyano cells.

Details

The indicators assigned to the "BS4BS5.Indicator" column in the full flowframe depends on the *interest* supplied. For *interest="bottom-right"* or *interest="top-right"*; 0 = Debris, 1 = BS4/BS5, 2 = not-identified while for *interest="Both"*, 0 = Debris, 1 = Syn-1, 2 = Syn-2, 3 = not-identified. This function calls the debris_nc or debris_inc function to identify debris and afterwards call the bs4_nc and/or bs5_nc depending on the interest supplied.

Value

list containing;

- **fullframe -** full flowframe with indicator for debris, BS4/BS5 or both.
- **reducedframe -** flowframe with onlySynechococcus cyanobacteria.
-
- **Cell_count -** a vector containing number of Synechococcus cyanobacteria cells. Might be a single value or vector of two values depending on interest.
- **Debris_count -** number of debris particles.

See Also

celldebris_emclustering

Examples

```
flowfile_path <- system.file("extdata", "B4_18_1.fcs", package = "cyanoFilter",
              mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
```

```
                                dataset = 1) #FCS file contains only one data object
flowfile_nona <- cyanoFilter::nona(x = flowfile)
flowfile_noneg <- cyanoFilter::noneg(x = flowfile_nona)
flowfile_logtrans <- cyanoFilter::lnTrans(x = flowfile_noneg, c('SSC.W', 'TIME'))
cells_nonmargin <- cellmargin(flow.frame = flowfile_logtrans, Channel = 'SSC.W',
           type = 'estimate', y_toplot = "FSC.HLin")
celldebris_nc(flowframe = cells_nonmargin$reducedframe, channel1 = "RED.B.HLin",
              channel2 = "YEL.B.HLin",
              interest = "bottom-right", to_retain = "refined")
```

`cellmargin` *Removes or assign indicators to margin events.*

Description

The function identifies margin events, i.e. cells that are too large for the flow cytometer to measure.

Usage

```
cellmargin(flow.frame, Channel = "SSC.W", type = c("manual",
  "estimate"), cut = NULL, y_toplot = "FSC,HLin")
```

Arguments

`flow.frame`	Flowframe containing margin events to be filtered out
`Channel`	The channel on which margin events are. Defaults to SSC.W (side scatter width)
`type`	The method to be used in gating out the margin cells. Can either be 'manual' where user supplies a cut off point on the channel, 1 = not margin 0 = margin
`cut`	sould not be NULL if type = 'manual'
`y_toplot`	channel on y-axis of plot with *Channel* used to gate out margin events

Details

Users can either supply a cut-off point along the channel describing particle width or allow the function to estimate the cut-off point using the `deGate` function from the *flowDensity* package. A plot of channel against "FSC.HLin" is provided with a vertical line showing the cut-off point separating margin events from other cells.

Value

list containing;

- **reducedflowframe -** flowframe without margin events
- **fullflowframe -** flowframe with an Margin.Indicator added as an extra column added to the expression matrix to indicate which particles are margin events. 1 = not margin event, 0 = margin event

- **N_margin -** number of margin events recorded
- **N_cell -** numner of non-margin events
- **N_particle -** is the number of particles in total, i.e. N_cell + N_margin

Examples

```
flowfile_path <- system.file("extdata", "B4_18_1.fcs", package = "cyanoFilter",
             mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
                               dataset = 1) #FCS file contains only one data object
flowfile_nona <- cyanoFilter::nona(x = flowfile)
flowfile_noneg <- cyanoFilter::noneg(x = flowfile_nona)
flowfile_logtrans <- lnTrans(x = flowfile_noneg, c('SSC.W', 'TIME'))
cellmargin(flow.frame = flowfile_logtrans, Channel = 'SSC.W',
           type = 'estimate', y_toplot = "FSC.HLin")
```

`cluster_plot` *plots the expression matrix of a flowframe analysed with celldebris_emclustering.*

Description

plots the expression matrix of a flowframe analysed with celldebris_emclustering.

Usage

```
cluster_plot(flowfile, channels, mus = NULL, tau = NULL, classifier)
```

Arguments

`flowfile`	flowframe to be plotted
`channels`	channels used in gating
`mus`	matrix of means obtained from celldebris_emclustering
`tau`	vector of cluster weights obtained from celldebris_emclustering
`classifier`	cells will be assigned to a cluster if belongs to that cluster by at least this probability. Only for plotting purposes.

cyanoFilter *cyanoFilter: A package to identify and/or assign indicators to BS4, BS5 cyanobacteria cells contained in water sample.*

Description

The package provides two categories of functions: *metafile* preprocessing functions and *fcsfile* processing functions.

metafile preprocessing functions

This set of functions (goodfcs and retain) helps to identify the appropriate fcs file to read.

fcsfile processing functions

These functions (nona and noneg, noneg, celldebris_nc, celldebris_emclustering) works on the fcs file to identify the cell populations contained in the sample that generated this file.

debris_inc *gates out or assign indicators to debris particle.*

Description

The function takes in a flowframe and identifies debris contained in the provided flowframe. It is specially designed for flowframe contaning both debris, BS4, BS5 and possibly other invading populations.

Usage

```
debris_inc(flowframe, p1, p2)
```

Arguments

flowframe	flowframe with debris and other cells.
p1	first flowcytometer channel that can be used to separate debris from the rest, e.g. "RED.B.HLin".
p2	second flowcytometer channel that can be used to separate debris from the rest, e.g. "YEL.B.HLin"

Details

The function uses the getPeaks and deGate functions in the flowDensity package to identify peaks between peaks and identify cut-off points between these peaks. A plot of both channels supplied with horizontal line separating debris from other cell populations is also returned.

Value

list containing;

- **syn - flowframe containing non-debris particles**
- **deb_pos - position of particles that are debris**
- **syn_pos - position of particles that are not debris**

See Also

debris_nc

Examples

```
flowfile_path <- system.file("extdata", "text.fcs", package = "cyanoFilter",
              mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
                               dataset = 1) #FCS file contains only one data object
flowfile_nona <- cyanoFilter::nona(x = flowfile)
flowfile_noneg <- cyanoFilter::noneg(x = flowfile_nona)
flowfile_logtrans <- lnTrans(x = flowfile_noneg, c('SSC.W', 'TIME'))
cells_nonmargin <- cellmargin(flow.frame = flowfile_logtrans, Channel = 'SSC.W',
           type = 'estimate', y_toplot = "FSC.HLin")
debris_inc(flowframe = flowfile, p1 = "RED.B.HLin", p2 = "YEL.B.HLin")
```

---

debris_nc *gates out or assign indicators to debris particle.*

Description

The function takes in a flowframe and identifies debris contained in the provided flowframe.

Usage

```
debris_nc(flowframe, p1, p2)
```

Arguments

flowframe	flowframe with debris and other cells.
p1	first flowcytometer channel that can be used to separate debris from the rest, e.g. "RED.B.HLin".
p2	second flowcytometer channel that can be used to separate debris from the rest, e.g. "YEL.B.HLin"

Details

The function uses the getPeaks and deGate functions in the flowDensity package to identify peaks between peaks and identify cut-off points between these peaks. A plot of both channels supplied with horizontal line separating debris from other cell populations is also returned.

Value

list containing;

- **syn - flowframe containing non-debris particles**
- **deb_pos - position of particles that are debris**
- **syn_pos - position of particles that are not debris**

See Also

debris_inc

---

goodfcs *indicates if measurement from a flowfile is good or bad.*

Description

This function examines the column containig $cells/\mu L$ and determins if the measurement can be used for further analysis or not based on a supplied range.

Usage

```
goodfcs(metafile, col_cpml = "CellspML", mxd_cellpML = 1000,
  mnd_cellpML = 50)
```

Arguments

metafile	associated metafile to the supplied fcsfile. This is a csv file containig computed stats from the flow cytometer.
col_cpml	column name or column number in metafile containing cell per microlitre measurements.
mxd_cellpML	maximal accepted cell per microlitre. Flowfiles with larger cell per microlitre are termed bad. Defaults to 1000.
mnd_cellpML	minimum accepted cell per microlitre. Flowfiles with lesser cell per microlitre are termed bad. Defaults to 50.

Details

Most flow cytometer makers will always inform clients within which range can measurements from the machine be trusted. The machines normally stores the amount of $cells/\mu L$ it counted in a sample. Too large value could mean possible doublets and too low value could mean too little cells.

Value

character vector with length same as the number of rows in the metafile whose entries are **good** for good files and **bad** for bad files.

Examples

```
metadata <- system.file("extdata", "2019-03-25_Rstarted.csv", package = "cyanoFilter",
             mustWork = TRUE)
metafile <- read.csv(metadata, skip = 7, stringsAsFactors = FALSE,
                     check.names = TRUE, encoding = "UTF-8")
metafile <- metafile[, 1:65] #first 65 columns contains useful information
#extract the part of the Sample.ID that corresponds to BS4 or BS5
metafile$Sample.ID2 <- stringr::str_extract(metafile$Sample.ID, "BS*[4-5]")
#clean up the Cells.muL column
names(metafile)[which(stringr::str_detect(names(metafile), "Cells."))] <- "CellspML"
goodfcs(metafile = metafile, col_cpml = "CellspML", mxd_cellpML = 1000, mnd_cellpML = 50)
```

---

lnTrans *log transforms the expression matrix of a flowframe*

Description

log transforms the expression matrix of a flowframe

Usage

```
lnTrans(x, notToTransform = c("SSC.W", "TIME"))
```

Arguments

x	flowframe to be transformed
notToTransform	columns not to be transformed

Value

flowframe with log transformed expression matrix

Examples

```
flowfile_path <- system.file("extdata", "B4_18_1.fcs", package = "cyanoFilter",
             mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
                               dataset = 1) #FCS file contains only one data object
flowfile_nona <- cyanoFilter::nona(x = flowfile)
flowfile_noneg <- cyanoFilter::noneg(x = flowfile_nona)
lnTrans(x = flowfile_noneg, c('SSC.W', 'TIME'))
```

mvnorm *multivariate normal density*

Description

multivariate normal density

Usage

```
mvnorm(x, mu, sigma)
```

Arguments

x	matrix to compute density on
mu	mean vector
sigma	variance covariance matrix

Value

vector of density

nona *Removes NA values from the expression matrix of a flow cytometer file.*

Description

Removes NA values from the expression matrix of a flow cytometer file.

Usage

```
nona(x)
```

Arguments

x	flowframe with expression matrix containing NAs.

Value

flowframe with expression matrix rid of NAs.

Examples

```
flowfile_path <- system.file("extdata", "B4_18_1.fcs", package = "cyanoFilter",
              mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
                               dataset = 1) #FCS file contains only one data object
nona(x = flowfile)
```

---

noneg *Removes negative values from the expression matrix*

Description

Removes negative values from the expression matrix

Usage

```
noneg(x)
```

Arguments

x	is the flowframe whose expression matrix contains negative values

Value

flowframe with non-negative values in its expression matrix

Examples

```
flowfile_path <- system.file("extdata", "B4_18_1.fcs", package = "cyanoFilter",
              mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
                               dataset = 1) #FCS file contains only one data object
flowfile_nona <- cyanoFilter::nona(x = flowfile)
noneg(x = flowfile_nona)
```

pair_plot *plots the expression matrix of a flowframe. Note that, it takes some time to display the plot.*

Description

plots the expression matrix of a flowframe. Note that, it takes some time to display the plot.

Usage

```
pair_plot(flowfile, notToPlot = c("TIME"))
```

Arguments

flowfile	flowframe to be plotted
notToPlot	column in expression matrix not to be plotted

Examples

```
flowfile_path <- system.file("extdata", "text.fcs", package = "cyanoFilter",
             mustWork = TRUE)
flowfile <- flowCore::read.FCS(flowfile_path, alter.names = TRUE,
                               transformation = FALSE, emptyValue = FALSE,
                               dataset = 1) #FCS file contains only one data object
flowfile_nona <- cyanoFilter::nona(x = flowfile)
flowfile_noneg <- cyanoFilter::noneg(x = flowfile_nona)
flowfile_logtrans <- lnTrans(x = flowfile_noneg, c('SSC.W', 'TIME'))
pair_plot(x = flowfile_logtrans,
          notToPlot = c("TIME", "FSC.HLin", "RED.R.HLin", "NIR.R.HLin"))
```

retain *Decides if a file should be retiained or removed based on its status.*

Description

Function to determine what files to retain and finally read from the flow cytometer FCS file.

Usage

```
retain(meta_files, make_decision = c("maxi", "mini", "unique"),
  Status = "Status", CellspML = "CellspML")
```

Arguments

meta_files	dataframe from meta file that has been preprocessed by the goodfcs function.
make_decision	decision to be made should more than one $cells/\mu L$ be good.
Status	column name in meta_files containing status obtained from the goodfcs function.
CellspML	column name in meta_files containing $cells/\mu L$ measurements.

Details

It is typically not known in advance which dilution level would result in the desired $cells/\mu L$, therefore the samples are ran through the flow cytometer at two or more dilution levels. Out of these, one has to decide which to retain and finally use for further analysis. This function and goodfcs are to help you decide that. If more than one of the dilution levels are judged good, the option *make_decision = "maxi"* will give "Retain" to the row with the maximum $cells/\mu L$ while the opposite occurs for *make_decision = "mini"*. *make_decision = "unique"* i there is only one measurement for that particular sample, while *make_decision = "maxi"* and *make_decision = "mini"* should be used for files with more than one measurement for the sample in question.

Value

a character vector with entries "Retain" for a file to be retained or "No!" for a file to be discarded.

See Also

goodfcs

Examples

```
metadata <- system.file("extdata", "2019-03-25_Rstarted.csv", package = "cyanoFilter",
             mustWork = TRUE)
metafile <- read.csv(metadata, skip = 7, stringsAsFactors = FALSE,
                     check.names = TRUE, encoding = "UTF-8")
metafile <- metafile[, 1:65] #first 65 columns contain useful information
#extract the part of the Sample.ID that corresponds to BS4 or BS5
metafile$Sample.ID2 <- stringr::str_extract(metafile$Sample.ID, "BS*[4-5]")
#clean up the Cells.muL column
names(metafile)[which(stringr::str_detect(names(metafile), "Cells."))] <- "CellspML"
metafile$Status <- cyanoFilter::goodfcs(metafile = metafile, col_cpml = "CellspML",
                           mxd_cellpML = 1000, mnd_cellpML = 50)
metafile$Retained <- NULL
# first 3 rows contain BS4 measurements at 3 dilution levels
metafile$Retained[1:3] <- cyanoFilter::retain(meta_files = metafile[1:3,], make_decision = "maxi",
                  Status = "Status", CellspML = "CellspML")
# last 3 rows contain BS5 measurements at 3 dilution levels as well
metafile$Retained[4:6] <- cyanoFilter::retain(meta_files = metafile[4:6,], make_decision = "maxi",
                  Status = "Status", CellspML = "CellspML")
```

Index