Package: CONFESS (via r-universe)

October 30, 2026

Type Package

Title Cell OrderiNg by FluorEScence Signal

Version 1.40.0

Date 2019-09-19

Author Diana LOW and Efthimios MOTAKIS

Maintainer Diana LOW <lowdiana@gmail.com>

Description Single Cell Fluidigm Spot Detector.

License GPL-2

LazyData TRUE

Depends R (>= 3.3),grDevices,utils,stats,graphics

Imports methods,changepoint,cluster,contrast,data.table(>= 1.9.7),ecp,EBImage,flexmix,flowCore,flowClust,flowMeans,flowMerge,flowPeaks,foreach,ggplot2,grid,limma,MASS,mor

biocViews ImmunoOncology, GeneExpression,DataImport,CellBiology,Clustering,RNASeq,QualityControl,Visualization,TimeCourse,Regression,Class

Collate fluo_est.R fluo_NBE.R internal_fluo_NBE.R internal_fluo_est.R internal_DDHF.R simulator.R cases.R

RoxygenNote 6.1.1

Encoding UTF-8

VignetteBuilder knitr

Suggests BiocStyle, knitr, rmarkdown, CONFESSdata

NeedsCompilation no

Config/pak/sysreqs cmake libfftw3-dev libgdal-dev gdal-bin libgeos-dev libgmp3-dev make libgsl0-dev libicu-dev libjpeg-dev libpng-dev libtiff-dev libuv1-dev libssl-dev libproj-dev libsqlite3-dev zlib1g-dev

Repository Bioconductor 3.23

Date/Publication 2026-04-28 12:43:14 UTC

RemoteUrl https://github.com/bioc/CONFESS

RemoteRef RELEASE_3_23

RemoteSha fe2e932367d1449a2e86901e53046c7ac3b8eeed

Contents

clu . . . 2
cluster2outlier . . . 4
createFluo . . . 5
defineLocClusters . . . 6
estimates . . . 7
estimates.2 . . . 9
files . . . 10
Fluo_adjustment . . . 11
Fluo_CV_modeling . . . 13
Fluo_CV_prep . . . 14
Fluo_inspection . . . 17
Fluo_modeling . . . 18
Fluo_ordering . . . 20
FluoSelection_byRun . . . 21
getFluo . . . 22
getFluo_byRun . . . 23
LocationMatrix . . . 24
pathEstimator . . . 25
readFiles . . . 26
Results . . . 27
simcells . . . 28
spotEstimator . . . 29
step1 . . . 32
step2 . . . 32
step2.1 . . . 35
step3 . . . 36
step3.1 . . . 37
step4 . . . 37
steps2_4 . . . 38

Index **40**

`clu` *clu*

Description

Example output from defineLoClusters

Usage

```
data("clu")
```

Format

The format is: List of 9 $ Results :'data.frame': 14 obs. of 15 variables: ..$ SampleID : chr [1:14] "1772-062-248_A01" "1772-062-248_A02" "1772-062-248_A03" "1772-062-248_A04" ... ..$ X : num [1:14] 259 491 262 261 261 258 259 189 498 194 ... ..$ Y : num [1:14] 367 219 368 369 335 367 336 278 20 284 ... ..$ Size : num [1:14] 31 49 19 152 141 43 59 15 49 32 ... ..$ Estimation.Type: chr [1:14] "Both.Channels" "Both.Channels" "One.Channel" "One.Channel" ... ..$ fore_Green : num [1:14] 48.4 36 26.2 45.7 32.6 ... ..$ back_Green : num [1:14] 17.2 17.3 16.6 16.9 17.1 ... ..$ fore_Red : num [1:14] 219.1 27.6 86.5 18.4 48 ... ..$ back_Red : num [1:14] 17.5 18.6 17.5 18.1 18 ... ..$ Green.StN : num [1:14] 1.442 1.01 0.626 1.389 0.889 ... ..$ Green.Pvalue : num [1:14] 6.03e-07 1.08e-03 5.55e-02 5.16e-27 4.57e-23 ... ..$ Red.StN : num [1:14] 3.5689 0.5455 2.2422 0.0256 1.3664 ... ..$ Red.Pvalue : num [1:14] 6.16e-07 2.68e-01 7.13e-05 1.00 3.33e-25 ... ..$ Other.Spots : chr [1:14] "0" "0" "X = 30, Y = 204 (Green) | X = 262, Y = 368 (Red)" "0" ... ..$ QCgroup : chr [1:14] "confidence" "outlier" "confidence" "confidence" ... $ BFdata :List of 14 ..$:List of 6 .. ..$ sample : chr "1772-062-248_A01" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A02" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A03" .. ..$ centerR: num [1:2] 263 370 .. ..$ centerG: num [1:2] 263 370 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-062-248_A04" .. ..$ centerR: num [1:2] 265 370 .. ..$ centerG: num [1:2] 265 370 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-062-248_A05" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A06" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A07" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A01" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A02" .. ..$ centerR: num [1:2] 195 250 .. ..$ centerG: num [1:2] 195 250 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A03" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A04" .. ..$ centerR: num [1:2] 191 281 .. ..$ centerG: num [1:2] 191 281 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A05" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A06" .. ..$ centerR: num [1:2] 187 274 .. ..$ centerG: num [1:2] 187 274 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A07" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL $ Processed.Files:List of 6 ..$ BF : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A01_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A02_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A03_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A04_BF.txt" ... ..$ CH1 : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A01_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A02_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A03_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A04_Green.txt" ... ..$ CH2 : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A01_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-

062-248_A02_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A03_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A04_Red.txt" ... ..$ separator : chr "_" ..$ image.type: chr [1:3] "BF" "Green" "Red" ..$ dateIndex : chr "WedApr611:21:282016" $ Outlier.indices: int [1:2] 2 9 $ Medians : chr [1:4, 1:4] "1772-062-248" "1772-062-248" "1772-067-039" "1772-067-039" ... $ Wellsets : chr [1:14, 1:3] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:3] "" "" "WellID" $ BFarea : num 7 $ image.type : chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedApr611:21:282016"

Value

example intermediates

`cluster2outlier` *cluster2outlier*

Description

It turns one or more selected clusters to outlier clusters, i.e. clusters consisting of outlying corrected signals.

Usage

```
cluster2outlier(data, out.cluster)
```

Arguments

`data`	List. The output of Fluo_inspection().
`out.cluster`	Numeric vector. The cluster number(s) to be turned into outlier clusters.

Value

A list of corrected fluorescence signal estimates with the selected clusters turned into outlier clusters.

Examples

```
### here we (erroneously) assume that cluster 1 is an outlier and we flag it so below
step3.withoutliers <- cluster2outlier(step3,out.cluster=1)

### the outlier samples can be removed by FluoSelection_byRun()
step3.withoutliers <- FluoSelection_byRun(step3.withoutliers,
                                  other=which(step3.withoutliers$GAPgroups[,1]!=-999))
```

createFluo *createFluo*

Description

The data format creator function for the signal normalization step.

Usage

```
createFluo(data, dateIndex = c(), from.file = FALSE, separator = "_")
```

Arguments

data	Data matrix. The output data matrix of LocationMatrix().
dateIndex	a date index to be used for storing the output files. It is either transfered from LocationMatrix() or it is generated here for the first time (e.g. if image analysis was not run by CONFESS or if the analysis has been repeated many times).
from.file	Logical. If TRUE the data is read from a file whose format should be the same to the output of LocationMatrix(). Default is FALSE.
separator	Character string. It separates the run ID from the Well ID in the image filenames (the «separator1» of readFiles()). It is also used here to enable the user perform the analysis independently of the previous step (cell recognition via imaging). Default is "_".

Value

A list of reformed data to be used in subsequent analysis: index: The sample indices. RGexprs: the foreground (columns 1 and 3) and background (columns 2 and 4) signals of each channel that have been estimated by spotEstimator() and filtered in LocationMatrix(). samples: the sample IDs. batch: a matrix of the run IDs. The first column contains the original run IDs. The second column is the converted original IDs into numeric values (to be used in the statistical modeling step of Fluo_adjustment()). size: the estimated cell size. image.type: the image type IDs as defined in readFiles(). The parameter is kept in ordeer to enable the user to use this function independently of the image analysis step. dateIndex: a date index to be used for storing the output files. It is either transfered from LocationMatrix() or it is generated here for the first time (e.g. if image analysis was not run by CONFESS or if the analysis has been repeated many times).

Examples

```
step1 <- createFluo(from.file=system.file("extdata", "Results_of_image_analysis.txt",
package = "CONFESS"),separator="_")
```

defineLocClusters *defineLocClusters*

Description

It performs quality check on the estimated location of spotEstimator() in order to flag possible outliers. The flagging is done both visually and statistically using the Grubbs test.

Usage

```
defineLocClusters(LocData, dims = rep(512, 2),
  out.method = "interactive.clustering", subset = c(),
  separator = "_", savePlot = "screen")
```

Arguments

LocData	The table of the location estimates obtained by spotEstimator().
dims	Numeric vector. The dimensions of the image data. Default is rep(512,2).
out.method	Character string. The method by which to flag outliers: "interactive.clustering" or "interactive.manual" or "manual". Default is "interactive.clustering". The interactive options work through interactive plots: "interactive.clustering" enables the user to highight the outliers via co-centric circles in the plot while "interactive.manual" asks the user to click on the plot to highlight the outliers (to confirm & finalize the picks in each plot the user has to select the "stop" command (Windows) or press the right click in Linux/Mac. Note that 'interactive.clustering' works when one has more then or equal to 15 samples IN EACH CATEGORY (Run/Well combination). The "manual" option simply gives back the original table of location estimates with the last column being a series of "confidence". The outliers should be manually annotated by inserting "outlier" in the appropriate rows of the last column.
subset	List. It allows the user to run the algorithm for a subset of data (run ids and wells). Default c() using all data. Otherwise put the run IDs and the wells (left and/or right) in a list, e.g. list(c("1772-115-xxx","1772-115-yyy"),"left").
separator	Character string. It refers to «separator1» parameter described in readFiles() that separates the run ID from the Well ID in the original image (converted) file names. Default is "_".
savePlot	Character string. Directory to store the plots if out.method = manual. Its value can be an existing directory or "screen" that prints the plot only on the screen. Default is the current working directory, getwd().

Details

The outlier locations will be re-estimated by BF image modelling or adjusted as the 2-dimensional median of all non-outlying locations.

Value

A list of components summarizing the location estimates and their quality control statistics: Results: The table of the location estimates from spotEstimator() with an extra "QCgroup" labelled column that flags the samples either by "confidence" or by "outlier" (the locations that have been selected as outliers from the interactive plots). If out.method = "manual" the column includes a series of "confidence" entries. The outliers should be manually labelled. BFdata: the outlier estimates of spotEstimator(). They are kept here for processing in the second spotEstimator() step. See spotEstimator() for more details. Processed.Files: the samples that have been processed by spotEstimator(). Also kept from the first spotEstimator() step. They will be processed in the second spotEstimator() step. Outlier.indices: a vector of outlier sample indices. They are generated from the flagging of the outliers via interactive plots. They have to be manually specified if out.method = "manual". Medians: the 2-dimenional medians by run ID and wellID sets. Wellsets: a matrix showing the directionality of the well IDs. BFarea: the size of the pseudospot. image.type: the image type IDs. dateIndex: a date index to be used in saving the output files.

Examples

```
library(CONFESSdata)
### set your directories
basedir<-"~/"
data_path<-system.file("extdata",package="CONFESSdata")
files<-readFiles(iDirectory=NULL,
                 BFdirectory=paste(data_path,"/BF",sep=""),
                 CHdirectory=paste(data_path,"/CH",sep=""),
                 separator = "_",image.type = c("BF","Green","Red"),
                 bits=2^16)

#this example is run using out.method="manual" (not interactive)
clu <- defineLocClusters(LocData=estimates,out.method="manual",savePlot="screen")
```

---

estimates *estimates*

Description

Example output of the SpotEstimator function

Usage

```
data("estimates")
```

Format

The format is: List of 6 $ SpotResults :'data.frame': 14 obs. of 14 variables: ..$ SampleID : chr [1:14] "1772-062-248_A01" "1772-062-248_A02" "1772-062-248_A03" "1772-062-248_A04" ... ..$ X : num [1:14] 259 491 262 261 261 258 259 189 498 194 ... ..$ Y : num [1:14] 367 219 368 369 335 367 336 278 20 284 ... ..$ Size : num [1:14] 31 49 19 152 141 43 59 15 49 32 ... ..$ Estimation.Type: chr [1:14] "Both.Channels" "Both.Channels" "One.Channel" "One.Channel" ... ..$

fore_Green : num [1:14] 48.4 36 26.2 45.7 32.6 ... ..$ back_Green : num [1:14] 17.2 17.3 16.6 16.9 17.1 ... ..$ fore_Red : num [1:14] 219.1 27.6 86.5 18.4 48 ... ..$ back_Red : num [1:14] 17.5 18.6 17.5 18.1 18 ... ..$ Green.StN : num [1:14] 1.442 1.01 0.626 1.389 0.889 ... ..$ Green.Pvalue : num [1:14] 6.03e-07 1.08e-03 5.55e-02 5.16e-27 4.57e-23 ... ..$ Red.StN : num [1:14] 3.5689 0.5455 2.2422 0.0256 1.3664 ... ..$ Red.Pvalue : num [1:14] 6.16e-07 2.68e-01 7.13e-05 1.00 3.33e-25 ... ..$ Other.Spots : chr [1:14] "0" "0" "X = 30, Y = 204 (Green) | X = 262, Y = 368 (Red)" "0" ... $ Outlier.Estimates:List of 14 ..$:List of 6 .. ..$ sample : chr "1772-062-248_A01" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A02" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A03" .. ..$ centerR: num [1:2] 263 370 .. ..$ centerG: num [1:2] 263 370 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-062-248_A04" .. ..$ centerR: num [1:2] 265 370 .. ..$ centerG: num [1:2] 265 370 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-062-248_A05" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A06" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A07" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A01" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A02" .. ..$ centerR: num [1:2] 195 250 .. ..$ centerG: num [1:2] 195 250 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A03" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A04" .. ..$ centerR: num [1:2] 191 281 .. ..$ centerG: num [1:2] 191 281 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A05" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A06" .. ..$ centerR: num [1:2] 187 274 .. ..$ centerG: num [1:2] 187 274 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A07" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL $ Processed.Files :List of 6 ..$ BF : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A01_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A02_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A03_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A04_BF.txt" ... ..$ CH1 : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A01_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A02_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A03_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A04_Green.txt" ... ..$ CH2 : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A01_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A02_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A03_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A04_Red.txt" ... ..$ separator : chr "_" ..$ image.type: chr [1:3] "BF" "Green" "Red" ..$ dateIndex : chr "WedApr611:21:282016" $ BFarea : num 7 $ image.type : chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedApr611:21:282016"

Value

example intermediates

`estimates.2` *estimates.2*

Description

Example output from the 2nd run of the spotEstimator function

Usage

```
data("estimates.2")
```

Format

The format is: List of 6 $ SpotResults :'data.frame': 14 obs. of 15 variables: ..$ SampleID : chr [1:14] "1772-062-248_A01" "1772-062-248_A02" "1772-062-248_A03" "1772-062-248_A04" ... ..$ X : num [1:14] 259 261 262 261 261 258 259 189 195 194 ... ..$ Y : num [1:14] 367 335 368 369 335 367 336 278 250 284 ... ..$ Size : num [1:14] 31 49 19 152 141 43 59 15 49 32 ... ..$ Estimation.Type: chr [1:14] "Fluorescence-based" "Chip.Pattern-based" "Fluorescence-based" "Fluorescence-based" ... ..$ fore_Green : num [1:14] 48.4 18.4 26.2 45.7 32.6 ... ..$ back_Green : num [1:14] 17.2 16.8 16.6 16.9 17.1 ... ..$ fore_Red : num [1:14] 219.1 19.8 86.5 18.4 48 ... ..$ back_Red : num [1:14] 17.5 17.8 17.5 18.1 18 ... ..$ Green.StN : num [1:14] 1.442 0.118 0.626 1.389 0.889 ... ..$ Green.Pvalue : num [1:14] 6.03e-07 1.00 5.55e-02 5.16e-27 4.57e-23 ... ..$ Red.StN : num [1:14] 3.5689 0.1416 2.2422 0.0256 1.3664 ... ..$ Red.Pvalue : num [1:14] 6.16e-07 1.00 7.13e-05 1.00 3.33e-25 ... ..$ Other.Spots : chr [1:14] "0" "0" "X = 30, Y = 204 (Green) | X = 262, Y = 368 (Red)" "0" ... ..$ QCgroup : chr [1:14] "confidence" "contamination" "confidence" "confidence" ... $ Outlier.Estimates:List of 14 ..$:List of 6 .. ..$ sample : chr "1772-062-248_A01" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A02" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A03" .. ..$ centerR: num [1:2] 263 370 .. ..$ centerG: num [1:2] 263 370 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-062-248_A04" .. ..$ centerR: num [1:2] 265 370 .. ..$ centerG: num [1:2] 265 370 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-062-248_A05" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A06" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-062-248_A07" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A01" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A02" .. ..$ centerR: num [1:2] 195 250 .. ..$ centerG: num [1:2] 195 250 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A03" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A04" .. ..$ centerR: num [1:2] 191 281 .. ..$ centerG: num [1:2] 191 281 .. ..$ arR :

NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A05" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL ..$:List of 6 .. ..$ sample : chr "1772-067-039_A06" .. ..$ centerR: num [1:2] 187 274 .. ..$ centerG: num [1:2] 187 274 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : chr "BF" ..$:List of 6 .. ..$ sample : chr "1772-067-039_A07" .. ..$ centerR: num [1:2] 0 0 .. ..$ centerG: num [1:2] 0 0 .. ..$ arR : NULL .. ..$ arG : NULL .. ..$ warn : NULL $ Processed.Files :List of 6 ..$ BF : chr [1:2] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A02_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-067-039_A02_BF.txt" ..$ CH1 : chr [1:2] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A02_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-067-039_A02_Green.txt" ..$ CH2 : chr [1:2] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A02_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-067-039_A02_Red.txt" ..$ separator : chr "_" ..$ image.type: chr [1:3] "BF" "Green" "Red" ..$ dateIndex : chr "WedApr611:21:282016" $ BFarea : num 7 $ image.type : chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedApr611:21:282016"

Value

example intermediates

`files` *files*

Description

Example output of readFiles with file definition and locations

Usage

```
data("files")
```

Format

The format is: List of 6 $ BF : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A01_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A02_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A03_BF.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/BF/1772-062-248_A04_BF.txt" ... $ CH1 : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A01_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A02_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A03_Green.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A04_Green.txt" ... $ CH2 : chr [1:14] "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A01_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A02_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A03_Red.txt" "/home/diana/R/x86_64-pc-linux-gnu-library/3.2/CONFESSdata/extdata/CH/1772-062-248_A04_Red.txt" ... $ separator : chr "_" $ image.type: chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedApr611:21:282016"

Examples

```
data(files)
```

`Fluo_adjustment` *Fluo_adjustment*

Description

A summary of the signal adjustment algorithms into a single function. It corrects the run effect (if any) and performs background adjustment for appropriately transformed data.

Usage

```
Fluo_adjustment(data, BGmethod = "normexp", maxMix = 3,
  single.batch.analysis = 1, transformation = "log", prior.pi = 0.1,
  flex.reps = 50, flexmethod = "BIC", savePlot = getwd(),
  seed = NULL)
```

Arguments

`data`	List. The output of createFluo().
`BGmethod`	Character string. The type of image background correction to be performed. One of "normexp" or "subtract". Default is "normexp".
`maxMix`	Integer. The maximum number of components to fit into the mixture of regressions model. If maxMix=1 or if the the optimal number of the estimated components is 1, the model reduces to the classical 2-way ANOVA. Default is 3.
`single.batch.analysis`	Integer. The baseline run against with the run effect correction is perfomred. Default is 1. If 0, each run is used as baseline iteratively and the final corrected data are obtained as the average of all corrections.
`transformation`	Character string. One of bc (Box-Cox), log, log10, asinh transforms applied to the data. Default is "log".
`prior.pi`	Float. The prior probability to accept a component. Default is 0.1.
`flex.reps`	Integer. The iterations of the Expectation-Maximization algorithm to estimate the flexmix model. Default is 50.
`flexmethod`	Character string. A method to estimate the optimal number of flexmix components. One of "BIC", "AIC", "ICL". Default is "BIC".
`savePlot`	Character string. Directory to store the plots. Its value can be an existing directory or "screen" that prints the plot only on the screen or "OFF" that does not generate a plot (suggested only during cross-validations). Default is the current working directory, getwd().
`seed`	Integer. An optional seed number for the Random Number Generator. Note that this seed is a 'reference' value of the actual seed used in sampling. CONFESS is using various random sampling methods. Each method's actual seed is factor*seed. The factors vary across methods. Default is NULL.

Value

A list with the data description, the normalized and corrected estimates over all runs by averaging (Summarized_estimates) AND for a particular "reference" run (Batch_estimates). Analytically, the components are: General index: The sample indices. samples: the sample IDs. batch: a matrix of the run IDs. The first column contains the original run IDs. The second column is the converted original IDs into numeric values (to be used in the statistical modeling step of Fluo_adjustment()). Size: the estimated cell size. RGexprs: the foreground (columns 1 and 3) and background (columns 2 and 4) signals of each channel that have been estimated by spotEstimator() and filtered in LocationMatrix(). exprs: the background corrected (only) signals of each channel. These data are fed into the flexmix model. image.type: the image type IDs as defined in readFiles(). dateIndex: the date index used. single.batch.analysis: the reference run used for run effect correction with flexmix. BGmethod: the background correction method used. maxMix: the maxMix parameter used. prior.pi: the prior.pi parameter used. flex.reps: the flex.reps parameter used. flexmethod: the flexmethod parameter used. RNG: the seed that is used to generate the results.

Summarized_estimates: corrected.exprs: the background and run effect corrected channel signals (by averaging the estimates of all runs). corrected.transformed.exprs: the background and run effect transformed corrected channel signals (by averaging the estimates of all runs). The transformation is defined in the transformation parameter (see above). allResults: the background and run effect corrected and transformed corrected channel signals (two different slots) for all runs.

Batch_estimates: it contains the analytical results for each batch in different slots. Each slot includes: corrected.exprs: the background and run effect corrected channel signals (for a run). corrected.transformed.exprs: the background and run effect transformed corrected channel signals (for a run). The transformation is defined in the transformation parameter (see above). mixes.(image.type 1): the estimated components of the flexmix model for one channel. mixes.(image.type 2): the estimated components of the flexmix model for the other channel. Batch.(image.type 1).est: the run effects of one channel. It contains the model estimates and significance P-values/FDRs. "Comp" corresponds to the factor of flexmix components (mixes) and "Batch" to the factor of runs. Batch.(image.type 2).est: the run effects of the other channel. It contains the model estimates and significance P-values/FDRs. "Comp" corresponds to the factor of flexmix components (mixes) and "Batch" to the factor of runs. fitted.values: the fitted values of the flexmix model for each channel. transformation: the transformation applied on the fluorescence signals (it stores the value of transformation parameter). model.residuals: the flexmix residuals for each channel. model.standardized.residuals: the flexmix standardized residuals for each channel. residual.statistics: the result of various normality tests for the residuals. lpar: the lambda parameter of the Box-Cox transformation (if used). design.(image.type 1): the design matrix of one channel. design.(image.type 2): the design matrix of the other channel. reference: the run that has been used as reference. (image.type 1).contrasts: the contrasts matrix for the differences across flexmix components and runs for one channel (only for the reference batch if any). (image.type 2).contrasts: the contrasts matrix for the differences across flexmix components and runs for the other channel (only for the reference batch if any).

Examples

```
step2 <- Fluo_adjustment(data=step1,flex.reps = 5,single.batch.analysis=5,savePlot="OFF")
```

Fluo_CV_modeling *Fluo_CV_modeling*

Description

It performs the cross-validation analysis on the estimated pseudotimes and clusters of the previous step, i.e. Fluo_CV_prep() or a manually generated list based on Fluo_modeling(). This function will evaluate the change in the estimated obtained (i) from a subset of data by f-fold cross-validation where f is the percentage of the samples from a specific group (@GAPgroups) that stay in the analysis at each CV iteration, or (ii) from a subset of runs that stay in the analysis at each CV iteration. It produces informative plots for the differences in the estimates between each iteration and the original estimates. It also summarizes the CV-estimated pseudotimes into a new set of estimates.

Usage

```
Fluo_CV_modeling(data, B = 20, batch = 1, perc.cutoff = 0.6,
  q = 0.9, f = 0.9, seed.it = TRUE, pseudotime.cutoff = 20,
  savePlot = getwd())
```

Arguments

data	List. The output of Fluo_CV_prep() or any other manually retrieved list with the components of Fluo_CV_prep().
B	Integer. The number of cross-validation to be performed. Default is 20.
batch	Numeric. A vector of runs to remain in the cross-validation. The rest are temporarily removed. The algorithm estimates the centroids of the reduced data and then calls the out-of-bag samples and re-estimates their k-mean clusters.
perc.cutoff	Float. The percentage of similar CV-estimated pseudotimes for each sample. The similarity is assessed by k-means with k = 2. It serves as a cut-off to identify outlying CV-estimated pseudotimes (along with q and pseudotime.cutoff). Default is 0.6.
q	Float. The q-th quantile of the difference between the original data estimated pseudotimes and the CV-estimated pseudotimes for each sample. It serves as a cut-off to identify outlying CV-estimated pseudotimes (along with perc.cutoff and pseudotime.cutoff). Default is 0.9.
f	Float. The percentage of samples from each estimated cluster (@GAPgroups) to remain in the cross-validation analysis. The rest are temporarily removed. The algorithm estimates the centroids of the reduced data and then calls the out-of-bag samples and re-estimates their k-mean clusters.
seed.it	Logical. If TRUE it performs cross-validation with the seed used in the analysis of the original data, i.e. in Fluo_CV_prep(). Default is TRUE.
pseudotime.cutoff	Integer. A user-defined value to define outlier samples (along with perc.cutoff and q), i.e. samples with Pseudotime(original) - medianPseudotime(CV) > pseudotime.cutoff. Default is 20.

savePlot — Character string. Directory to store the plots of the analysis of the whole data. Its value can be an existing directory or "screen" that prints the plot only on the screen. The "OFF" option is permanently used in cross-validations). Default is the current working directory, getwd().

Value

The output of Fluo_modeling() with the original estimates and the CV-based estimated pseudotimes/clusters in different slots of component CV results. The results are categorized by run number. Each run contains the original estimates (@Original Pseudotimes), the CV-based estimates by the "median/original" method (@Reest.Pseudotimes_median/original) and the CV-based estimates by the "median/null" method (@Reest.Pseudotimes_median/null).

1. "median/original" It integrates the information of the CV and the originally estimated pseudotimes. It build kmean clusters of the B CV estimates for each sample and defines pseudotime(i) = median(pseudotime(set1,i)) where set1 is a subset of the B pseudotimes that exhibit some similarity. The similarity is assessed by k-means clustering. This subset should contain a large percentage of the B data (>perc.cutoff) and it's median should be lower than the q-th quantile of the average differences between the original and the CV-estimated pseudotimes across all samples. If the CV estimated pseudotimes do not satisfy the above then the algorithm returns pseudotime(i) = median(pseudotime(set2,i)) where set2 is the cluster of B pseudotimes that minimizes |median(pseudotimes(set2,i))-original.pseudotimes|.

2. "median/null" if set1 with similar pseudotimes that satisfies the above rules exists, it returns the pseudotime(i) = median(pseudotime(set1,i)). Otherwise it returns NULL, i.e. the sample CV-estimated pseudotimes are not similar and the algorithm cannot estimate reliably the pseudotime of interest.

Both solutions are then going under a final round of change-point analysis that uses the CV-estimated pseudotimes and produce the final results of Fluo_CV_modeling(). All results canbe subsequently used in Fluo_ordering(). The output also includes a second component, @All.Progressions, with the original and the CV estimated pseudotimes. This information is kept for comparison reasons and it is not used further.

Examples

```
print("Not run because takes a long time")
#step1 <- createFluo(from.file=system.file("extdata", "Results_of_image_analysis.txt",
#package = "CONFESS"),separator="_")
#steps2_4 <- Fluo_CV_prep(data=step1,init.path = "bottom/left",path.type=c("circular","clockwise"),
#single.batch.analysis = 5,flex.reps=5,altFUN="kmeans",VSmethod="DDHFmv",CPmethod="ECP",
#B.kmeans=5,CPpvalue=0.01,savePlot="OFF")
#steps2_4cv<-Fluo_CV_modeling(data=steps2_4,B=5,f=0.99,savePlot="OFF")
```

Fluo_CV_prep *Fluo_CV_prep*

Description

It generates the data that will be used in the cross-validation analysis. Essentialy, it analyzes and stores the original (full) dataset for different reference runs, seeds, starting clusters etc. It estimates the progression path automatically that is feasible only for standard paths (path.type parameter different than 'other'). For this reason this function is useful only in these cases. If otherwise, it should be ommitted from the analysis and the user is should generate it manually, i.e. run Fluo_adjustment() - Fluo_modeling() series as many times as the cases to be studied with manual init.path input in Fluo_modeling().

Usage

```
Fluo_CV_prep(data, init.path = "bottom/left", path.type = c("circular",
  "clockwise"), BGmethod = "normexp", maxMix = 3,
  single.batch.analysis = 1:5, transformation = "log",
  prior.pi = 0.1, flex.reps = 50, flexmethod = "BIC", areacut = 0,
  fixClusters = 0, altFUN = "kmeans", k.max = 15,
  VSmethod = "DDHFmv", CPmethod = "ECP", CPgroups = 5,
  B.kmeans = 50, CPpvalue = 0.05, CPmingroup = 15,
  savePlot = getwd(), seed = NULL)
```

Arguments

data	List. The output of crearteFluo(), i.e. the image analysis estimates.
init.path	Character vector. It defines the starting cluster of the progression path in general terms. It can be one of "top/right", "top/left", "bottom/right" or "bottom/left" indicating the cluster of interest on the 2d scatterplot of Fluo_inspection(). Default is rep("bottom/left",2), i.e. in Fucci an EM/earlyG1 like cluster.
path.type	Character vector. A user-defined vector that characterizes the cell progression dynamics. The first element can be either "circular" or "A2Z" or "other". If "circular" the path progression is assummed to exhibit a circle-like behavior. If "A2Z" the path is assumed to have a well-defined start and a well-defined end point (e.g. a linear progression). If "other" the progression is assumed to be arbitrary without an obvious directionality. Default is "circular". The second element can be either "clockwise" or "anticlockwise" depending on how the path is expected to proceed. Default is "clockwise". If the first element is "other" the second element can be ommited. If path.type = "other", the function does not estimate a path. The cross-validation algorithm will probably fail for this kind of path.type values because it will not be able to automatically guess the progression path. It is suggested that the user runs the cross-validation manually (each time specifying the path in Fluo_modeling()), collect the data in a list similar to the one produced here and input them into Fluo_CV_modeling() to get the results.
BGmethod	Character string. The type of image background correction to be performed. One of "normexp" or "subtract". Default is "normexp".
maxMix	Integer. The maximum number of components to fit into the mixture of regressions model. If maxMix=1 or if the the optimal number of the estimated components is 1, the model reduces to the classical 2-way ANOVA. Default is 3.

`single.batch.analysis`
: Numeric. The baseline run(s) to perform run effect correction with flexmix. Due to iterative nature of this function it can be a series of values includying 0 (averaging of run correction estimates). Default is 1:5.

`transformation`
: Character string. One of bc (Box-Cox), log, log10, asinh transforms applied to the data. Default is "log".

`prior.pi`
: Float. The prior probability to accept a component. Default is 0.1.

`flex.reps`
: Integer. The iterations of the Expectation-Maximization algorithm to estimate the flexmix model. Default is 50.

`flexmethod`
: Character string. A method to estimate the optimal number of flexmix components. One of "BIC", "AIC", "ICL". Default is "BIC".

`areacut`
: Integer. The "artificial" area size (BFarea^2) of the cells estimated by BF image modelling. Default is 0, implying that the area sizes to be corrected will by estimated automatically from the data (not recommended if prior knowledge exists).

`fixClusters`
: Integer. A number that defines the number of k-mean clusters to be initially generated. If 0, the function runs GAP analysis to estimate the optimal number of clusters. Default is 0.

`altFUN`
: Character string. A user-defined method to generate the initial clusters. It can be one of kmeans, samSpec, fmeans,fmerge or fpeaks. Default is "kmeans".

`k.max`
: Integer. This is the maximum number of clusters that can be generated by k-means (if fixClusters = 0). Default is 15.

`VSmethod`
: Character string. The variance stabilization transformation method to be applied to the corrected fluorescence data prior to the change point analysis. IT can be one of "log" or "DDHFmv". Default is "DDHFmv".

`CPmethod`
: Character string. The change point method to be used. It can be one of "ECP", (non-parametric) "manualECP" (non-parametric with user-defined numner of change-points) or "PELT" (Pruned Exact Linear Time; parametric). Default is ECP.

`CPgroups`
: Integer. The number of change-points to be kept if CPmethod = "manualECP". Default is 5.

`B.kmeans`
: Integer. The number of bootstrap samples for the calculation of the GAP statistic. Default is 50.

`CPpvalue`
: Float. The significance level below which we do not reject a change point. Default is 0.05.

`CPmingroup`
: Integer. The minimum number of values for a cluster re-estimated by the change-point analysis. Default is 10.

`savePlot`
: Character string. Directory to store the plots of the analysis of the whole data. Its value can be an existing directory or "screen" that prints the plot only on the screen. The "OFF" option is permanently used in cross-validations). Default is the current working directory, getwd().

`seed`
: Integer. An optional seed number for the Random Number Generator. Note that this seed is a 'reference' value of the actual seed used in sampling. CONFESS is using various random sampling methods. Each method's actual seed is factor*seed. The factors vary across methods. Default is NULL.

Details

The function can also be used to generate all pseudotime/clustering results up to the function of Fluo_modeling() but the starting cluster has to be defined in general terms (see init.path parameter below). For this reason, its parameters are essentially the same to the ones defined previously at the Fluo_adjustment() - Fluo_modeling() functions.

Value

The results of Fluo_modeling() for difference reference runs (batches) are stored in different slots. An additional slot @init.path exists that stores the init.path parameter (its value to be used in the CV automatically).

One can directly use the run components in Fluo_ordering() to finalize the data analysis. The main purpose of this function, though, is to prepare the data for cross-validation.

Examples

```
step1 <- createFluo(from.file=system.file("extdata", "Results_of_image_analysis.txt",
package = "CONFESS"),separator="_")
steps2_4 <- Fluo_CV_prep(data=step1,init.path = "bottom/left",path.type=c("circular","clockwise"),
single.batch.analysis = 5,flex.reps=5,altFUN="kmeans",VSmethod="DDHFmv",CPmethod="ECP",
B.kmeans=5,CPpvalue=0.01,savePlot="OFF")
```

Fluo_inspection *Fluo_inspection*

Description

It generates the initial cell clusters as defined by their corrected fluorescence signals. The clusters can be generated by k-means (with GAP statistic estimated number of clusters) or by flow cytometry based approaches. This function shows the number and the characteristics of the initial groups and help us inspect cells' progression type for pathEstimator().

Usage

```
Fluo_inspection(data, altFUN = "kmeans", fixClusters = 0,
  SAM.sigma = 200, k.max = 15, B.kmeans = 50, savePlot = getwd(),
  seed = NULL)
```

Arguments

data	List. The output of getFluo() or getFluo_byRun().
altFUN	Character string. A user-defined method to generate the initial clusters. It can be one of kmeans, samSpec, fmeans,fmerge or fpeaks. Default is "kmeans".
fixClusters	Integer. A number that defines the number of k-mean clusters to be initially generated. If 0, the function runs GAP analysis to estimate the optimal number of clusters. Default is 0.

SAM.sigma	Integer. A value for the sigma parameter of SamSPECTRAL algorithm. Default is 200.
k.max	Integer. This is the maximum number of clusters that can be generated by k-means (if fixClusters = 0). Default is 15.
B.kmeans	Integer. The number of bootstrap samples for the calculation of the GAP statistic. Default is 50.
savePlot	Character string. Directory to store the plots. Its value can be an existing directory or "screen" that prints the plot only on the screen or "OFF" that does not generate a plot (suggested only during cross-validations). Default is the current working directory, getwd().
seed	Integer. An optional seed number for the Random Number Generator. Note that this seed is a 'reference' value of the actual seed used in sampling. CONFESS is using various random sampling methods. Each method's actual seed is factor*seed. The factors vary across methods. Default is NULL.

Value

A list of corrected fluorescence signal estimates and a helper plot for deciding the number of groups and the cell progression path. The output is essentially the output of getFluo() or getFluo_byRun() with the addition of the following components: GAPgroups: the groups estimated by one of the altFUN methods are depicted in the first column. The second column contains 1s for non-outlier signals and 2s for outlier signals (as estimated by each of the methods). clusterFUN: the altFUN method that has been used for clustering. normal.sigma: the sigma parameter of samSpec method. centroids: the 2 dimensional medians (centroids) of the estimated clusters. fixClusters: the fixClusters parameter used. Kmax: the k.meax parameter used. B.kmeans: the B.kmeans parameter used

Examples

```
step3 <- Fluo_inspection(data=step2.1,altFUN="kmeans",B.kmeans=5,savePlot="OFF")
```

Fluo_modeling *Fluo_modeling*

Description

It takes the initial groups and the path progression and estimates the pseudotimes of cell progression and the associated change-points (updated cell clusters).

Usage

```
Fluo_modeling(data, init.path, VSmethod = "DDHFmv", CPmethod = "ECP",
  CPgroups = 5, CPpvalue = 0.05, CPmingroup = 10, seed = NULL)
```

Arguments

data	List. The output of pathEstimator().
init.path	Numeric vector. The cell path progression as it has been estimated by pathEstimator() or a user-defined path that can be deduced from Fluo_inspection(). The latter is suggested only when path.type = "other" in pathEstimator().
VSmethod	Character string. The variance stabilization transformation method to be applied to the corrected fluorescence data prior to the change point analysis. IT can be one of "log" or "DDHFmv". Default is "DDHFmv".
CPmethod	Character string. The change point method to be used. It can be one of "ECP", (non-parametric) "manualECP" (non-parametric with user-defined numner of change-points) or "PELT" (Pruned Exact Linear Time; parametric). Default is ECP.
CPgroups	Integer. The number of change-points to be kept if CPmethod = "manualECP". Default is 5.
CPpvalue	Float. The significance level below which we do not reject a change point. Default is 0.05.
CPmingroup	Integer. The minimum number of values for a cluster re-estimated by the change-point analysis. Default is 10.
seed	Integer. An optional seed number for the Random Number Generator. Note that this seed is a 'reference' value of the actual seed used in sampling. CONFESS is using various random sampling methods. Each method's actual seed is factor*seed. The factors vary across methods. Default is NULL.

Value

A list of corrected fluorescence signal estimates, the pseudotimes and the cell progression clusters. The output is essentially the output of pathEstimator() with the addition of the following components: UpdatedPath: the updated progression path after re-estimation by change points and clustering. DataSorts: a matrix contains the calculated distances by orthogonal projection and the pseudotimes. DDHFupdate: it takes TRUE or FALSE to signify whether the clustering/pseudotime estimation has been updated by the re-estimation procedure. corrected.VStransformed.exprs: the background and run effect transformed corrected channel signals (by one of "log" or "DDHFmv"). The transformation is defined in the VSmethod parameter. VSmethod: the transformation that has been applied to the channel signals. Progression: it describes the estimated progression by the pseudotimes (first column) and the differences between the transformed channel signals. Updated.groups: the final clusters. CPs: the final change points detected. CPmethod: the CPmethod parameter used. CPsig: the CPpvalue parameter used. CPgroups: the CPgroups parameter used. CPmingroup: the CPmingroup parameter used.

Examples

```
step4<-Fluo_modeling(data=step3.1,init.path=step3.1$Path,VSmethod="DDHFmv",
                     CPmethod="ECP",CPpvalue=0.01)
```

Fluo_ordering *Fluo_ordering*

Description

It produces the final output table of CONFESS. It includes the Sample IDs, the Run IDs, the estimated cell areas (image analysis), the corrected fluorescence signals of both channels (run and background adjustED), the pseudotimes of cell progression, the final cell clusters and other statistics of cell progression analysis.

Usage

```
Fluo_ordering(data, den.method = "wavelets", savePlot = "OFF")
```

Arguments

data	List. The outut of Fluo_modeling().
den.method	Character string. A method to denoise the transformed channel signal differences (used for change-point analysis). The denoising obtains the residuals that can be subjected to statistical testing (model assumptions). It is one of "splines", "wavelets" or "lregr" (linear regression). Default is "wavelets".
savePlot	Character string. Directory to store the plots. Its value can be an existing directory or "screen" that prints the plot only on the screen or "OFF" that does not generate a plot (suggested only during cross-validations). Default is the current working directory, getwd().

Value

The list of final results in two components: Summary_results: It contains a matrix that summarizes the findings of CONFESS. It has the index number of each sample, the sample IDs, the run IDs, the estimated cell size, the estimated run corrected cell size, the estimated pseudotime, the log, and if specified, DDHFmv transformed channel signals, the log or DDHFmv transformed channel differences, the estimated clusters, the residuals and a column flagginf outlier samples.

Analytical results: It contains all the components of Fluo_modeling() with the addition of: Outliers: a vector having "normal" for non-outlier samples and "outlier" for outlier samples. The outliers are estimated by Grubbs statistic based on their distance from the bulk of the clustered samples. Residuals: the residuals of the fitted model for the denoising of the corrected transformed channel differences (see parameter den.method). Residuals_diagnostics: various normality tests for the estimated residuals.

The component of

Examples

```
step5<-Fluo_ordering(data=step4,savePlot="OFF")
```

FluoSelection_byRun *FluoSelection_byRun*

Description

It accepts a subset of data to inspect their background corrected fluorescence signal characteristics. Typically it one can inout the data from a single run to identify an appropriate mixture model for run effect correction. Any other arbitrary subset of the data can also be used. For example, it can be used to keep certain samples and filter out outliers.

Usage

```
FluoSelection_byRun(data, batch = c(), other = c())
```

Arguments

data	List. The output of createFluo().
batch	Integer. A selected run. If it is c() then the "other" parameter should be activated. Default is 1.
other	Numeric vector. It accepts the sample numbers indicating the samples to be kept for analysis, e.g. other = c(1:10, 101:110) to keep samples 1:10 and 100:110. Default is c().

Value

A list of reformed data to be used in subsequent analysis. It is essentially the same slots of createFluo() with only a subset of data included (as defined by the batch and other parameters): index: The sample indices. RGexprs: the foreground (columns 1 and 3) and background (columns 2 and 4) signals of each channel that have been estimated by spotEstimator() and filtered in LocationMatrix(). samples: the sample IDs. batch: a matrix of the run IDs. The first column contains the original run IDs. The second column is the converted original IDs into numeric values (to be used in the statistical modeling step of Fluo_adjustment()). size: the estimated cell size. image.type: the image type IDs as defined in readFiles(). The parameter is kept in ordeer to enable the user to use this function independently of the image analysis step. dateIndex: a date index to be used for storing the output files. It is either transfered from LocationMatrix() or it is generated here for the first time (e.g. if image analysis was not run by CONFESS or if the analysis has been repeated many times).

Examples

```
step1 <- createFluo(from.file=system.file("extdata", "Results_of_image_analysis.txt",
                  package = "CONFESS"),separator="_")
step2a <- FluoSelection_byRun(data = step1, batch = 4:5)
```

getFluo *getFluo*

Description

It retrieves the run effect and background corrected signals.

Usage

```
getFluo(data, areacut = 0)
```

Arguments

data	List. The output of the Fluo_adjustment().
areacut	Integer. The "artificial" area size (BFarea^2) of the cells estimated by BF image modelling. Default is 0, implying that the area sizes to be corrected will by estimated automatically from the data (not recommended if prior knowledge exists).

Value

A list of estimates to be used in subsequent analysis (the slots are the same to those of getFluo_byRun()): index: The sample indices. samples: the sample IDs. batch: a matrix of the run IDs. The first column contains the original run IDs. The second column is the converted original IDs into numeric values (to be used in the statistical modeling step of Fluo_adjustment()). Size: the estimated cell size. corrected.exprs: the background corrected channel signals (case of a single run). corrected.transformed.exprs: the background transformed corrected channel signals (case of a single run). The transformation is defined in the transformation parameter. correctedAreas: the log-transformed areas after correction and imputation. areacut: the above areacut if different from 0 or the automatically calulated one otherwise. transformation: the transformation applied on the fluorescence signals. image.type: the image type IDs as defined in readFiles(). The parameter is kept in order to enable the user to use this function independently of the image analysis step. dateIndex: the date index used. single.batch.analysis: the reference run of the run effect correction by flexmix. BGmethod: the background correction methods used. maxMix: the maxMix parameter used. prior.pi: the prior.pi parameter used. flex.reps: the flex.reps parameter used. flexmethod: the flexmethod parameter used. RNG: the seed that is used to generate the results.

Examples

```
step1 <- createFluo(from.file=system.file("extdata", "Results_of_image_analysis.txt",
package = "CONFESS"),separator="_")
step2.1 <- getFluo(data=step2)
```

getFluo_byRun *getFluo_byRun*

Description

It produces the background corrected data when run correction is not needed. It can be used for data coming from a single run instead of Fluo_adjustment(). Alternatively, this function can be used to visualize the fluorescence densities of a single batch before deciding the form of the normalization model.

Usage

```
getFluo_byRun(data, BGmethod = "normexp", areacut = 0,
  transformation = "log", savePlot = getwd())
```

Arguments

data	List. The output of createFluo().
BGmethod	Character string. The type of image background correction to be performed. One of "normexp" or "subtract". Default is "normexp".
areacut	Integer. The "artificial" area size (BFarea^2) of the cells estimated by BF image modelling. Default is 0, implying that the area sizes to be corrected will by estimated automatically from the data (not recommended if prior knowledge exists).
transformation	Character string. One of bc (Box-Cox), log, log10, asinh transforms applied to the data. Default is "log".
savePlot	Character string. Directory to store the plots. Its value can be an existing directory or "screen" that prints the plot only on the screen or "OFF" that does not generate a plot (suggested only during cross-validations). Default is the current working directory, getwd().

Value

A list of corrected signal estimates. The slots are the same to those of getFluo(): index: The sample indices. samples: the sample IDs. batch: a matrix of the run IDs. The first column contains the original run IDs. The second column is the converted original IDs into numeric values (to be used in the statistical modeling step of Fluo_adjustment()). Size: the estimated cell size. corrected.exprs: the background corrected channel signals (case of a single run). corrected.transformed.exprs: the background transformed corrected channel signals (case of a single run). The transformation is defined in the transformation parameter. correctedAreas: the log-transformed areas after correction and imputation. areacut: the above areacut if different from 0 or the automatically calulated one otherwise. transformation: the transformation applied on the fluorescence signals. image.type: the image type IDs as defined in readFiles(). The parameter is kept in order to enable the user to use this function independently of the image analysis step. dateIndex: the date index used. single.batch.analysis: it returns 0 because there is no run effect correction done. BGmethod: the background correction methods used. maxMix: it returns NULL because there is no flexmix run

effect correction done. prior.pi: it returns NULL because there is no flexmix run effect correction done. flex.reps: it returns NULL because there is no flexmix run effect correction done. flexmethod: it returns NULL because there is no flexmix run effect correction done. RNG: the seed that is used to generate the results.

Examples

```
step1 <- createFluo(from.file=system.file("extdata", "Results_of_image_analysis.txt",
package = "CONFESS"),separator="_")

### select the samples of a single run and correct them
step2a <- FluoSelection_byRun(data = step1, batch = 5)
step2.1 <- getFluo_byRun(data=step2a,savePlot="OFF")
```

`LocationMatrix` *LocationMatrix*

Description

It generates the final cell location and fluorescnece signal estimates and summarizes the quality control statistics.

Usage

```
LocationMatrix(data, filter.by = matrix(c("FDR", "Out.Index", 0.005,
  "confidence"), ncol = 2), report.by.signif = "max")
```

Arguments

`data`	Data matrix. The matrix of the location and fluorescence signal estimates after two rounds (maximum) of spotEstimator().
`filter.by`	Data matrix. A series of filtering criteria and cut-offs that specify which samples are KEPT for further analysis (see vignette). By default it flags by FDR (alpha = 0.005) and outlier index (keeps only the 'confident' estimates).
`report.by.signif`	Character string. It returns the pre-defined channel-specific signal-to-noise ratio and test statistics for each sample. If "min", the algorithm only reports the P-values/FDRs and signal-to-noise of the channel with the minimum signal-to-noise ratio. If "max", the algorithm only reports the P-values/FDRs and signal-to-noise of the channel with the maximum signal-to-noise ratio. Default is "max".

Value

List. The first component is a data matrix of the final table of estimates. The main body of this table has been generated by spotEstimator(). It summarizes the location, the raw fluorescence signal estimates (foreground and background) and the quality control statistics. It keeps only the signal-to-noise ratio and the associated P-value/FDR of a predefined channel (see parameter report.by.signif).

The last column ("Cells") consists of 1s for the samples that pass the filtering step (filter.by) and are used for further analysis. The rest of the samples are assigned 0s. The user should always inspect them along with the images to obtain the final list of samples to be used for further analysis. The second component is the date index for storing the output files. It is transfered to the next step.

Examples

```
### the results matrix (column 'Cells') indicates three empty capture chambers
### (thus not only outliers were associated with the absense of a cell!)
Results <- LocationMatrix(data=estimates.2,
           filter.by = matrix(c("FDR","Out.Index",0.005,"confidence"),ncol=2))
```

`pathEstimator` *pathEstimator*

Description

It reads the generated groups of Fluo_inspection() and estimates the path cell progression given a user-defined expected pattern. It can also join some of the groups into a single one (manual selection is required).

Usage

```
pathEstimator(data, path.start = 1, path.type = c("circular",
  "clockwise"), joinedGroups = NULL)
```

Arguments

`data`	List. The output of Fluo_inspection().
`path.start`	Integer. A cluster number indicating the starting cluster that algorithm should use to build the path. The cluster numbers refer to the plot generated by Fluo_inspection(). Default is 1. If path.type = "circular" the number does not matter. If path.type = "A2Z" the user should inspect the Fluo_inspection() plot to detect the beginning of the path. If path.type = "other", the function will not estimate a path. The user has to manually insert the path progression (the cluster numbers) in Fluo_modeling().
`path.type`	Character vector. A user-defined vector that characterizes the cell progression dynamics. The first element can be either "circular" or "A2Z" or "other". If "circular" the path progression is assummed to exhibit a circle-like behavior. If "A2Z" the path is assumed to have a well-defined start and a well-defined end point (e.g. a linear progression). If "other" the progression is assumed to be arbitrary without an obvious directionality. Default is "circular". The second element can be either "clockwise" or "anticlockwise" depending on how the path is expected to proceed. Default is "clockwise". If the first element is "other" the second element can be ommited. If path.type = "other", the function does not estimate a path. The exact path has to be manually inserted in Fluo_modeling().

joinedGroups — List. A list of cluster numbers to join. E.g. list(c(2,4)) joins cluster 2 and 4 as depicted in the Fluo_inspection() plot. Alternatively, list(c(2,4),c(1,6)) joins cluster 2 and 4 and clusters 1 and 6 as depicted in the Fluo_inspection() plot.Each list entry should contain 2 groups. Default is NULL.

Value

The list of adjusted signal estimates, a progression path and the defined path type. The output is essentially the output of Fluo_inspection() with the addition of the following components: Path: the estimated path (visualized in the Fluo_Inspection() helper plot). path.type: the path.type that has been used to estimate the path.

Examples

```
step3.1 <- pathEstimator(step3,path.start=6,path.type=c("circular","clockwise"))
```

---

readFiles *readFiles*

Description

Reads the image data that are going to be analyzed. It converts the images into txt files. The images should be in .C01 (high resolution) or .BMP, or .JPG or .PNG format. The file names should be of the form:

Usage

```
readFiles(iDirectory, BFdirectory, CHdirectory, separator = "_",
  image.type = c("BF", "Red", "Green"), bits = 2^16)
```

Arguments

iDirectory — Character string. The directory where all images are stored. The images should be in the same format. Available choices are: C01, BMP, JPEG and PNG. The function recognizes the format automatically. If ommitted, the function assumes that the txt data already exist at the predefined folders.

BFdirectory — Character string. The directory to store the .txt converted Bright Field images.

CHdirectory — Character string. The directory to store the .txt converted channel (e.g. Red/Green) images

separator — Character string. This is the «separator2» parameter that removes the Bright Field ("BF") and channel indicators (IDs) from the image file names. Default is "_".

image.type — Character string. A triplet of IDs to characterize the type of images under study. They refer to the ImageType part of the original image or txt file names. Default is c("BF","Red","Green").

bits — Numeric. The image bits. It is used to unnormalize the C01 signals from readCellomics(). It does not affect the signals of other image types. Default is 2^16.

Details

"RunID(separator1)WellID(separator2)ImageType.ImageFormat

For example in "1772-062-248_A01@BF.C01", RunID = 1772-062-248", separator1 = _, WellID = A01, separator2 = @ ImageType = BF, ImageFormat = C01. The function expects to see both Bright Field and channel images. It will store them in different directories. It will return a list of the respective .txt file names. Note that separator1 and separator2 CAN BE the same character (e.g. "_").

If the images have been already converted, then the txt files should be stored in the above form with ImageFormat = txt.

readFiles() will take the minimum overlapping sets. Converted images not present in any of the channels or the Bright Field list will be reported and discarded.

Value

A list with the followign components: BF: the files names of the Bright Field converted data matrices. CH1: the files names of the converted data matrices of one channel. CH2: the files names of the converted data matrices of the other channel. separator: the separator being used. image.type: the image type IDs. dateIndex: a date index to be used in saving the output files.

Examples

```
library(CONFESSdata)

### set your directories
basedir<-"~/"
data_path<-system.file("extdata",package="CONFESSdata")

## to read txt files
files<-readFiles(iDirectory=NULL,
                 BFdirectory=paste(data_path,"/BF",sep=""),
                 CHdirectory=paste(data_path,"/CH",sep=""),
                 separator = "_",image.type = c("BF","Green","Red"),
                 bits=2^16)

## to convert from BMP/JPEG images
#write_dir<-"~/converted_images/"
#files<-readFiles(iDirectory=data_path,
#                 BFdirectory=paste(write_dir,"/BF",sep=""),
#                 CHdirectory=paste(write_dir,"/CH",sep=""),
#                 separator = "_",image.type = c("BF","Green","Red"),
#                 bits=2^16)
```

`Results` *Results*

Description

Example output from LocationMatrix

Usage

```
data("Results")
```

Format

The format is: List of 2 $ Output :'data.frame': 14 obs. of 15 variables: ..$ SampleID : chr [1:14] "1772-062-248_A01" "1772-062-248_A02" "1772-062-248_A03" "1772-062-248_A04" ... ..$ X : num [1:14] 259 261 262 261 261 258 259 189 195 194 ... ..$ Y : num [1:14] 367 335 368 369 335 367 336 278 250 284 ... ..$ Size : num [1:14] 31 49 19 152 141 43 59 15 49 32 ... ..$ Estimation.Type: chr [1:14] "Fluorescence-based" "Chip.Pattern-based" "Fluorescence-based" "Fluorescence-based" ... ..$ fore_Green : num [1:14] 48.4 18.4 26.2 45.7 32.6 ... ..$ back_Green : num [1:14] 17.2 16.8 16.6 16.9 17.1 ... ..$ fore_Red : num [1:14] 219.1 19.8 86.5 18.4 48 ... ..$ back_Red : num [1:14] 17.5 17.8 17.5 18.1 18 ... ..$ Signal-to-Noise: Factor w/ 14 levels "0.141617112045591",..: 12 1 8 5 4 7 10 6 3 11 ... ..$ Pvalue : Factor w/ 14 levels "0.000360562783835169",..: 13 7 14 11 9 12 8 1 5 10 ... ..$ FDR : Factor w/ 12 levels "0.0001684989130688",..: 9 6 1 7 11 10 12 2 6 8 ... ..$ Out.Index : Factor w/ 2 levels "confidence","contamination": 1 2 1 1 1 1 1 1 1 1 ... ..$ Other.Spots : Factor w/ 3 levels "0","X = 128, Y = 358 (Green) | X = 191, Y = 277 (Red)",..: 1 1 3 1 1 1 1 1 1 1 ... ..$ Cells : num [1:14] 1 0 1 1 1 1 1 1 0 1 ... $ dateIndex: chr "WedApr611:21:282016"

Value

example intermediates

---

simcells *simcells*

Description

The main function to simulate spots of various numbers, sizes, signals in one or multiple images of a given dimension.

Usage

```
simcells(channels = 2, spots.per.image = c(1, 1),
  one.location = c(50, 50), image.dimension = rep(100, 2),
  signal.level = list(700, 700), noise.level = c(200, 200),
  spot.size = list(30, 30), agreement.number = 1)
```

Arguments

channels	Integer. The number of channels for each sample. Default is 2.
spots.per.image	Numeric vector. The number of spots in each image (channel). The length of the vector equals to the number of channels. Default is one spot per channel.
one.location	Numeric vector. The central location of the matched spots across the channels (in pixel) coordinates. Default is (X,Y) = (50,50).

`image.dimension`
: Numeric vector. The image dimension (in pixels). Default is 100 x 100.

`signal.level`
: List. The lambda parameter of the Poisson distribution that generates the true spot (pixel) signals. The list has as many components (length) as the number of channels. The number of elements of each component equals to the number of spots in each particular channel. Default is list(700,700).

`noise.level`
: Numeric vector. The sigma parameter of the Normal distribution that generates the image noise level. The length of the vector equals to the number of channels. Default is c(200,200).

`spot.size`
: List. The size of each spot on each channel (in pixels). The list has as many components (length) as the number of channels. The number of elements of each component equals to the number of spots in each particular channel. Default is list(30,30).

`agreement.number`
: Integer. It defines how many spot pairs are matched, i.e. they are located in the same coordinates across channels. These reflect true cells. Default is 1 corresponding to a single-cell case study.

Value

The image(s) with the generated spot(s). It consists of the data matrices and the location of the spot centers.

Examples

```
r<-simcells(channels = 2, spots.per.image = c(2, 3), one.location = c(50, 50),
image.dimension = rep(200, 2), signal.level = list(c(1000, 1000), c(1000, 700, 300)),
noise.level = c(100, 100),spot.size = list(c(81, 100), c(26, 29, 50)), agreement.number = 1)

r<-simcells(channels = 2, spots.per.image = c(0, 0), image.dimension = rep(200, 2),
signal.level = list(c(),c()),noise.level = c(0, 0), spot.size = list(c(), c()))
```

`spotEstimator` *spotEstimator*

Description

The main function to produce the raw fluorescence signal estimation results by analysis of the Fluidigm images.

Usage

```
spotEstimator(files, correctionAlgorithm, subset = c(),
  foregroundCut = seq(0.5, 0.7, 0.02), denoise = FALSE,
  despeckle = FALSE, chip.type = "medium/large", cutSides = 0,
  BFarea = 7, log.transform = TRUE, minDiff = 0.5,
  show.possible.contamination = TRUE, cutoff = 50, QCdata = 0,
  median.correction = TRUE, savePlot = getwd())
```

Arguments

files	Character string. The file names to be read and analyzed. This is the output of readFiles()
correctionAlgorithm	Logical. Its value specifies the estimation stage. If FALSE, the function processes all data using the standard operations of spotCoords(), i.e. case detection and fluorescence signal estimation. This is the first estimation stage. If TRUE, the function processes the BF image modeling estimates of outlier images obtained by defineLocClusters(). The BF image modeling is internally applied during the first stage. Note that correctionAlgorithm = TRUE is strictly used in the second (outliers adjustment / correction) stage of the process.
subset	Numeric vector. It can be a series sample index numbers (a subset) that specifies the samples to be analyzed. The index numbers are obtained from readFiles() (the position of the sample in each listed vector). By default subset = c(). The parameter is mainly used in the second estimation stage where spotEstimator() processes the outlier images (the index numbers
foregroundCut	Numeric vector. The binary segmentation image analysis cutoffs for normalized image data. Pixels with normalized signals higher than the cutoff belong to foreground. Default is seq(0.5,0.7,0.02).
denoise	Logical. If TRUE it denoises the channel images with la8, universal, hard. Default is FALSE.
despeckle	Logical. If TRUE the bf image is descpeckled in the ImageJ fashion. Default is FALSE.
chip.type	Character string. It specifies the type of Fluidigm chip to be analyzed. Default is "medium/large". The alternative option is "small".
cutSides	Integer. It instructs the algorithm to find spots in a certain central image area. For example, for a 512 x 512 image with cutSides = 50, spotEstimator() will search for spots in the central area [cutSides:(512-cutSides),cutSides:(512-cutSides)] of the image matrix. Default is 0.
BFarea	Integer. Defines a rectangular pseudo-spot size whose fluorescence will be estimated. This is mainly used in BF image modeling where a fluorescence spot could not be originally detected. The value of this parameter is also used as a cut-off to find matched spots across channel of the same sample image. Default is 7.
log.transform	Logical. If TRUE the image data are plotted in the log scale. Default is TRUE
minDiff	Float. The mu_hat of the H0: image-to-noise ratio = log(foreground_signal) - log(background_signal) = mu_hat. Rejection of H0 implies that the identified spot is brighter than background. Default is 0.5.
show.possible.contamination	Logical. If TRUE it reports all identified unmatched spots in both channels. Default is TRUE.
cutoff	Integer. A cutoff of the distance between the estimated spot location of an outlier sample (X, Y) and the median location of all non-outliers of the same run and well set (medX,medY), i.e. (X-medX, Y-medY). An outlier sample can either have a fluorescence-based location (X, Y) or a BF-based location (X*,

Y*) or both. It is re-adjusted as follows: (1) if min(X-medX, Y-medY) > cutoff and min(X*-medX, Y*-medY) > cutoff, the sample's location is set to (medX, medY); (2) if min(X*-medX, Y*-medY) <= cutoff, the sample's location is set to (X*, Y*); (3) if min(X-medX, Y-medY) <= cutoff and min(X*-medX, Y*-medY) > cutoff, the algorithm can either produce the solution of (1) or the solution of (2) depending on the value of median.correction parameter below. By default cutoff = 50.

`QCdata` List. The output of defineLocClusters().

`median.correction`
Logical. If TRUE, the algorithm re-adjusts the location of the outlier sample as the median of all non-outliers of the same run and well ID (if necessary).

`savePlot` Character string. Directory to store the plots. Its value can be an existing directory or "screen" that prints the plot only on the screen. Default is the current working directory, getwd().

Details

Triplets of images of the same sample are sequentially considered to estimate the channel-specific fluorescence signals (if detectable) or perform BF image modeling. The main result of this function is a table of location and fluorescence estimates for each sample.

Value

A list of the following components: SpotResults: the matrix of the location and fluorescence signal estimates. It contains the index number of each sample, the X,Y coordinates of the spot center, the spot size, the type of estimation that have been performed (fluorescence based indicating the channels in which the spot has been found or BF image modelling based), the fluorescence foreground and background signals of each channel, the signal-to-noise ratio (logForeground - logBackground) for each channel, the associated P-value of significance of the signal-to-noise ratio and a column indicating the coordinates of other spots that are not matched in both images. Existence of such spots (values that are different from 0) indicate contaminated image or highly noisy images or images with other artefacts. If correctionAlgorithm=TRUE (second spotEstimator() step), there is an extra column generated indicating outlier samples (see the QCgroup column in defineLocClusters()). Outlier.Estimates: The estimates obtained from BF modeling (if necessary to be obtained). These are alternative location estimates that will be used if the original estimates of the SpotResults table are flagged as outliers. Processed.Files: the samples that have been processed by spotEstimator(). BFarea: the pseudospot size. image.type: the image type IDs. dateIndex: a date index to be used in saving the output files.

Examples

```
### set your directories
basedir<-"~/"
#data_path<-system.file("extdata",package="CONFESSdata")
#files<-readFiles(iDirectory=NULL,
#                 BFdirectory=paste(data_path,"/BF",sep=""),
#                 CHdirectory=paste(data_path,"/CH",sep=""),
#                 separator = "_",image.type = c("BF","Green","Red"),
#                 bits=2^16)
```

```
### an example where the second image produces a clear outlier!
#estimates <- spotEstimator(files=files,subset=1:3,foregroundCut=seq(0.6,0.76,0.02),
#                            correctionAlgorithm=FALSE,savePlot="screen")
```

---

step1 *step1*

Description

Example output of the createFluo function

Usage

```
data("step1")
```

Format

The format is: List of 7 $ index : int [1:246] 1 2 3 4 5 6 7 8 9 10 ... $ RGexprs :'data.frame': 246 obs. of 4 variables: ..$ fore_Green: num [1:246] 48.4 26.2 45.7 34 24 ... ..$ back_Green: num [1:246] 17.2 17 16.8 17.1 16.8 ... ..$ fore_Red : num [1:246] 219.1 86.5 18.4 84.4 104.3 ... ..$ back_Red : num [1:246] 17.3 17 18 17.7 17.5 ... $ samples : chr [1:246] "1772-062-248_A01" "1772-062-248_A03" "1772-062-248_A04" "1772-062-248_A06" ... $ batch : chr [1:246, 1:2] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... $ Size : num [1:246] 31 19 152 43 59 21 72 81 31 56 ... $ image.type: chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedMar2313:29:522016"

Value

example intermediates

---

step2 *step2*

Description

Example output of the Fluo_adjustment function

Usage

```
data("step2")
```

Format

The format is: List of 3 $ General :List of 15 ..$ index : int [1:246] 1 2 3 4 5 6 7 8 9 10 ... ..$ samples : chr [1:246] "1772-062-248_A01" "1772-062-248_A03" "1772-062-248_A04" "1772-062-248_A06" ... ..$ batch : chr [1:246, 1:2] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... ..$ Size : num [1:246] 31 19 152 43 59 21 72 81 31 56 ... ..$ RGexprs :'data.frame': 246 obs. of 4 variables: .. ..$ fore_Green: num [1:246] 48.4 26.2 45.7 34 24 ... .$ back_Green: num [1:246] 17.2 17 16.8 17.1 16.8 ... .$ fore_Red : num [1:246] 219.1 86.5 18.4 84.4 104.3 ... .$ back_Red : num [1:246] 17.3 17 18 17.7 17.5 ... ..$ exprs : num [1:246, 1:2] 35.3 13.2 32.9 20.9 11.3 ... .- attr(*, "dimnames")=List of 2 .. .. ..$: NULL .. .. ..$: chr [1:2] "Green" "Red" ..$ image.type : chr [1:3] "BF" "Green" "Red" ..$ dateIndex : chr "WedMar2313:29:522016" ..$ single.batch.analysis: num 5 ..$ BGmethod : chr "normexp" ..$ maxMix : num 3 ..$ prior.pi : num 0.1 ..$ flex.reps : num 5 ..$ flexmethod : chr "BIC" ..$ RNG : NULL $ Summarized_estimates:List of 3 ..$ corrected.exprs : num [1:246, 1:2] 37 11.8 34.4 20.5 9.7 ... .- attr(*, "dimnames")=List of 2 .. .. ..$: NULL .. .. ..$: chr [1:2] "Green" "Red" ..$ corrected.transformed.exprs: num [1:246, 1:2] 3.61 2.47 3.54 3.02 2.27 ... .- attr(*, "dimnames")=List of 2 .. .. ..$: NULL .. .. ..$: chr [1:2] "Green" "Red" ..$ allResults :List of 2 .. ..$ corrected.exprs : chr [1:247, 1:10] "Ref=1" "40.7543386916705" "13.2747695514339" "37.8669946332554" ... .$ corrected.transformed.exprs: chr [1:247, 1:10] "Ref=1" "3.70756230498048" "2.58586520738665" "3.63407987867093" ... $ Batch_estimates :List of 5 ..$ Batch1:List of 15 .. ..$ corrected.exprs : num [1:246, 1:2] 40.8 13.3 37.9 22.8 10.8 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ corrected.transformed.exprs : num [1:246, 1:2] 3.71 2.59 3.63 3.13 2.38 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ mixes.Green : num [1:246] 1 1 1 1 1 1 2 1 2 1 ... .$ mixes.Red : num [1:246] 3 3 1 3 3 3 3 3 3 3 ... .$ Batch.Green.est : chr [1:11, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Batch)2" ... .$ BatchRed.est : chr [1:16, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Comp)3" ... .$ fitted.values : num [1:246, 1:2] 2.7 2.7 2.7 2.7 2.7 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ transformation : chr "log" .. ..$ model.residuals : num [1:246, 1:2] 0.865 -0.114 0.797 0.34 -0.277 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ model.standardized.residuals: num [1:246, 1:2] 1.746 -0.231 1.609 0.687 -0.559 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ residual.statistics : chr [1:4, 1:7] "" "Green" "Red" "Green & Red" ... .$ lpar : NULL .. ..$ design.Green : num [1:246, 1:10] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:10] "(Intercept)" "factor(Comp)2" "factor(Batch)2" "factor(Batch)3" ... .- attr(*, "assign")= int [1:10] 0 1 2 2 2 2 3 3 3 3 .. .. ..- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ design.Red : num [1:246, 1:15] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:15] "(Intercept)" "factor(Comp)2" "factor(Comp)3" "factor(Batch)2" ... .- attr(*, "assign")= int [1:15] 0 1 1 2 2 2 2 3 3 3 ... .- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ reference : int 1 ..$ Batch2:List of 15 .. ..$ corrected.exprs : num [1:246, 1:2] 34.33 9.63 31.69 17.88 8.08 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ corrected.transformed.exprs : num [1:246, 1:2] 3.54 2.26 3.46 2.88 2.09 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ mixes.Green : num [1:246] 1 1 1 1 1 1 1 1 2 1 ... .$ mixes.Red : num [1:246] 3 3 1 3 3 3 3 3 3 3 ... .$ Batch.Green.est : chr [1:11, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Batch)1" ... .$ BatchRed.est : chr [1:16, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Comp)3" ... .$ fitted.values : num [1:246, 1:2] 2.85 2.85 2.85 2.85 2.85 ... .- attr(*, "dimnames")=List of 2 .. .. ..

..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ transformation : chr "log" .. ..$ model.residuals : num [1:246, 1:2] 0.717 -0.262 0.649 0.192 -0.425 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ model.standardized.residuals: num [1:246, 1:2] 1.345 -0.492 1.218 0.361 -0.797 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ residual.statistics : chr [1:4, 1:7] "" "Green" "Red" "Green & Red" ... .$ lpar : NULL .. ..$ design.Green : num [1:246, 1:10] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:10] "(Intercept)" "factor(Comp)2" "factor(Batch)1" "factor(Batch)3" ... .- attr(*, "assign")= int [1:10] 0 1 2 2 2 2 3 3 3 3 .. .. ..- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ design.Red : num [1:246, 1:15] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:15] "(Intercept)" "factor(Comp)2" "factor(Comp)3" "factor(Batch)1" ... .- attr(*, "assign")= int [1:15] 0 1 1 2 2 2 2 3 3 3 ... .- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ reference : int 2 ..$ Batch3:List of 15 .. ..$ corrected.exprs : num [1:246, 1:2] 41.7 12.6 38.7 22.7 10.1 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ corrected.transformed.exprs : num [1:246, 1:2] 3.73 2.53 3.66 3.12 2.31 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ mixes.Green : num [1:246] 1 1 1 1 1 1 2 1 2 1 ... .$ mixes.Red : num [1:246] 3 3 1 3 3 3 3 3 3 3 ... .$ Batch.Green.est : chr [1:11, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Batch)1" ... .$ BatchRed.est : chr [1:16, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Comp)3" ... .$ fitted.values : num [1:246, 1:2] 2.7 2.7 2.7 2.7 2.7 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ transformation : chr "log" .. ..$ model.residuals : num [1:246, 1:2] 0.865 -0.114 0.797 0.34 -0.277 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ model.standardized.residuals: num [1:246, 1:2] 1.749 -0.231 1.612 0.688 -0.56 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ residual.statistics : chr [1:4, 1:7] "" "Green" "Red" "Green & Red" ... .$ lpar : NULL .. ..$ design.Green : num [1:246, 1:10] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:10] "(Intercept)" "factor(Comp)2" "factor(Batch)1" "factor(Batch)2" ... .- attr(*, "assign")= int [1:10] 0 1 2 2 2 2 3 3 3 3 .. .. ..- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ design.Red : num [1:246, 1:15] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:15] "(Intercept)" "factor(Comp)2" "factor(Comp)3" "factor(Batch)1" ... .- attr(*, "assign")= int [1:15] 0 1 1 2 2 2 2 3 3 3 ... .- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ reference : int 3 ..$ Batch4:List of 15 .. ..$ corrected.exprs : num [1:246, 1:2] 34.67 11.85 32.27 19.75 9.81 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ corrected.transformed.exprs : num [1:246, 1:2] 3.55 2.47 3.47 2.98 2.28 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ mixes.Green : num [1:246] 1 1 1 1 1 1 2 1 2 1 ... .$ mixes.Red : num [1:246] 3 3 1 3 3 3 3 3 3 3 ... .$ Batch.Green.est : chr [1:11, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Batch)1" ... .$ BatchRed.est : chr [1:16, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Comp)3" ... .$ fitted.values : num [1:246, 1:2] 2.7 2.7 2.7 2.7 2.7 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ transformation : chr "log" .. ..$ model.residuals : num [1:246, 1:2] 0.865 -0.114 0.797 0.34 -0.277 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ model.standardized.residuals: num [1:246, 1:2] 1.741 -0.23 1.605 0.685 -0.557 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL ..

.. .. ..$: chr [1:2] "Green" "Red" .. ..$ residual.statistics : chr [1:4, 1:7] "" "Green" "Red" "Green & Red" ... .$ lpar : NULL .. ..$ design.Green : num [1:246, 1:10] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:10] "(Intercept)" "factor(Comp)2" "factor(Batch)1" "factor(Batch)2" ... .- attr(*, "assign")= int [1:10] 0 1 2 2 2 2 3 3 3 3 .. .. ..- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ design.Red : num [1:246, 1:15] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:15] "(Intercept)" "factor(Comp)2" "factor(Comp)3" "factor(Batch)1" ... .- attr(*, "assign")= int [1:15] 0 1 1 2 2 2 2 3 3 3 ... .- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ reference : int 4 ..$ Batch5:List of 17 .. ..$ corrected.exprs : num [1:246, 1:2] 34.3 11.96 31.95 19.7 9.95 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ corrected.transformed.exprs : num [1:246, 1:2] 3.54 2.48 3.46 2.98 2.3 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ mixes.Green : num [1:246] 1 1 1 1 1 1 2 1 2 1 ... .$ mixes.Red : num [1:246] 2 2 1 2 2 2 2 2 2 2 ... .$ Batch.Green.est : chr [1:11, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Batch)1" ... .$ BatchRed.est : chr [1:11, 1:4] "" "(Intercept)" "factor(Comp)2" "factor(Batch)1" ... .$ fitted.values : num [1:246, 1:2] 2.7 2.7 2.7 2.7 2.7 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ transformation : chr "log" .. ..$ model.residuals : num [1:246, 1:2] 0.865 -0.114 0.797 0.34 -0.277 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ model.standardized.residuals: num [1:246, 1:2] 1.749 -0.231 1.612 0.688 -0.56 ... .- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" .. ..$ residual.statistics : chr [1:4, 1:7] "" "Green" "Red" "Green & Red" ... .$ lpar : NULL .. ..$ design.Green : num [1:246, 1:10] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:10] "(Intercept)" "factor(Comp)2" "factor(Batch)1" "factor(Batch)2" ... .- attr(*, "assign")= int [1:10] 0 1 2 2 2 2 3 3 3 3 .. .. ..- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ design.Red : num [1:246, 1:10] 1 1 1 1 1 1 1 1 1 1 ... .- attr(*, "dimnames")=List of 2 .. ..$: chr [1:246] "1" "2" "3" "4" ... .$: chr [1:10] "(Intercept)" "factor(Comp)2" "factor(Batch)1" "factor(Batch)2" ... .- attr(*, "assign")= int [1:10] 0 1 2 2 2 2 3 3 3 3 .. .. ..- attr(*, "contrasts")=List of 2 .. ..$ factor(Comp) : chr "contr.treatment" .. ..$ factor(Batch): chr "contr.treatment" .. ..$ reference : int 5 .. ..$ Green.contrasts : chr [1:21, 1:7] "Channel" "CH1" "CH1" "CH1" ... .$ Red.contrasts : chr [1:21, 1:7] "Channel" "CH2" "CH2" "CH2" ...

Value

example intermediates

`step2.1` *step2.1*

Description

Example output of the getFluo function

Usage

```
data("step2.1")
```

Format

The format is: List of 18 $ index : int [1:246] 1 2 3 4 5 6 7 8 9 10 ... $ samples : chr [1:246] "1772-062-248_A01" "1772-062-248_A03" "1772-062-248_A04" "1772-062-248_A06" ... $ batch : chr [1:246, 1:2] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... $ Size : num [1:246] 31 19 152 43 59 21 72 81 31 56 ... $ corrected.exprs : num [1:246, 1:2] 34.3 11.96 31.95 19.7 9.95 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ corrected.transformed.exprs: num [1:246, 1:2] 3.54 2.48 3.46 2.98 2.3 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ correctedAreas : num [1:246] 3.43 2.94 5.02 3.76 4.08 ... $ areacut : num 49 $ transformation : chr "log" $ image.type : chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedMar2313:29:522016" $ single.batch.analysis : num 5 $ BGmethod : chr "normexp" $ maxMix : num 3 $ prior.pi : num 0.1 $ flex.reps : num 5 $ flexmethod : chr "BIC" $ RNG : NULL

Value

example intermediates

---

step3 *step3*

Description

Example output of the Fluo_inspection function

Usage

```
data("step3")
```

Format

The format is: List of 25 $ index : int [1:246] 1 2 3 4 5 6 7 8 9 10 ... $ samples : chr [1:246] "1772-062-248_A01" "1772-062-248_A03" "1772-062-248_A04" "1772-062-248_A06" ... $ batch : chr [1:246, 1:2] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... $ Size : num [1:246] 31 19 152 43 59 21 72 81 31 56 ... $ corrected.exprs : num [1:246, 1:2] 34.3 11.96 31.95 19.7 9.95 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ corrected.transformed.exprs: num [1:246, 1:2] 3.54 2.48 3.46 2.98 2.3 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ correctedAreas : num [1:246] 3.43 2.94 5.02 3.76 4.08 ... $ areacut : num 49 $ transformation : chr "log" $ image.type : chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedMar2313:29:522016" $ single.batch.analysis : num 5 $ BGmethod : chr "normexp" $ maxMix : num 3 $ prior.pi : num 0.1 $ flex.reps : num 5 $ flexmethod : chr "BIC" $ RNG : NULL $ GAPgroups : num [1:246, 1:2] 3 6 2 6 6 6 4 6 5 4 ... $ clusterFUN : chr "kmeans" $ normal.sigma : num 200 $ centroids : num [1:6, 1:3] 1 2 3 4 5 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:3] "Cluster" "Green" "Red" $ fixClusters : num 0 $ Kmax : num 15 $ B.kmeans : num 5

Value

example intermediates

`step3.1` *step3.1*

Description

Example output of the pathEstimator function

Usage

```
data("step3.1")
```

Format

The format is: List of 27 $ index : int [1:246] 1 2 3 4 5 6 7 8 9 10 ... $ samples : chr [1:246] "1772-062-248_A01" "1772-062-248_A03" "1772-062-248_A04" "1772-062-248_A06" ... $ batch : chr [1:246, 1:2] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... $ Size : num [1:246] 31 19 152 43 59 21 72 81 31 56 ... $ corrected.exprs : num [1:246, 1:2] 34.3 11.96 31.95 19.7 9.95 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ corrected.transformed.exprs: num [1:246, 1:2] 3.54 2.48 3.46 2.98 2.3 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ correctedAreas : num [1:246] 3.43 2.94 5.02 3.76 4.08 ... $ areacut : num 49 $ transformation : chr "log" $ image.type : chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedMar2313:29:522016" $ single.batch.analysis : num 5 $ BGmethod : chr "normexp" $ maxMix : num 3 $ prior.pi : num 0.1 $ flex.reps : num 5 $ flexmethod : chr "BIC" $ RNG : NULL $ GAPgroups : num [1:246, 1:2] 3 6 2 6 6 6 4 6 5 4 ... $ clusterFUN : chr "kmeans" $ normal.sigma : num 200 $ centroids : num [1:6, 1:3] 1 2 3 4 5 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:3] "Cluster" "Green" "Red" $ fixClusters : num 0 $ Kmax : num 15 $ B.kmeans : num 5 $ Path : num [1:6] 6 3 5 4 2 1 $ Path.type : chr [1:2] "circular" "clockwise"

Value

example intermediates

`step4` *step4*

Description

Example output of the Fluo_modeling function

Usage

```
data("step4")
```

Format

The format is: List of 39 $ index : int [1:246] 1 2 3 4 5 6 7 8 9 10 ... $ samples : chr [1:246] "1772-062-248_A01" "1772-062-248_A03" "1772-062-248_A04" "1772-062-248_A06" ... $ batch : chr [1:246, 1:2] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... $ Size : num [1:246] 31 19 152 43 59 21 72 81 31 56 ... $ corrected.exprs : num [1:246, 1:2] 34.3 11.96 31.95 19.7 9.95 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ corrected.transformed.exprs : num [1:246, 1:2] 3.54 2.48 3.46 2.98 2.3 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Green" "Red" $ correctedAreas : num [1:246] 3.43 2.94 5.02 3.76 4.08 ... $ areacut : num 49 $ transformation : chr "log" $ image.type : chr [1:3] "BF" "Green" "Red" $ dateIndex : chr "WedMar2313:29:522016" $ single.batch.analysis : num 5 $ BGmethod : chr "normexp" $ maxMix : num 3 $ prior.pi : num 0.1 $ flex.reps : num 5 $ flexmethod : chr "BIC" $ RNG : NULL $ GAPgroups : num [1:246, 1:2] 3 6 2 6 6 6 4 6 5 4 ... $ clusterFUN : chr "kmeans" $ normal.sigma : num 200 $ centroids : num [1:7, 1:3] 2 1 5 4 3 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:3] "Cluster" "Green" "Red" $ fixClusters : num 0 $ Kmax : num 15 $ B.kmeans : num 5 $ Path : num [1:6] 6 3 5 4 2 1 $ Path.type : chr [1:2] "circular" "clockwise" $ UpdatedPath : num [1:7] 1 2 3 4 5 6 7 $ DataSorts : chr [1:246, 1:2] "0.453392432450554" "0.0087217837649943" "0.201631879709111" "-0.395060882232867" ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Distance" "Pseudotime" $ DDHFupdate : logi FALSE $ corrected.VStransformed.exprs: num [1:246, 1:2] 93.7 87.7 90.3 89.3 87.7 ... $ VSmethod : chr "DDHFmv" $ Progression : num [1:246, 1:2] 77 18 201 22 31 11 169 34 119 171 ... ..- attr(*, "dimnames")=List of 2 .. ..$: NULL .. ..$: chr [1:2] "Pseudotime" "transf.Difference" $ Updated.groups : num [1:246] 2 1 5 1 1 1 4 1 3 4 ... $ CPs : num [1:6] 56 93 140 178 206 229 $ CPmethod : chr "ECP" $ CPsig : num 0.01 $ CPgroups : num 5 $ CPmingroup : num 10

Value

example intermediates

`steps2_4` *steps2_4*

Description

example results of the Fluo_CV_modeling function

Usage

```
data("steps2_4")
```

Format

The format is: List of 2 $ Batch5 :List of 39 ..$ index : int [1:246] 1 2 3 4 5 6 7 8 9 10 ... ..$ samples : chr [1:246] "1772-062-248_A01" "1772-062-248_A03" "1772-062-248_A04" "1772-062-248_A06" ... ..$ batch : chr [1:246, 1:2] "1772-062-248" "1772-062-248" "1772-062-248" "1772-062-248" ... ..$ Size : num [1:246] 31 19 152 43 59 21 72 81 31 56 ... ..$ corrected.exprs : num [1:246, 1:2] 26.64 9.5 24.84 15.43 8.04 ... .- attr(*, "dimnames")=List of 2 .. .. ..$:

NULL .. .. ..$: chr [1:2] "Green" "Red" ..$ corrected.transformed.exprs : num [1:246, 1:2] 3.28 2.25 3.21 2.74 2.08 ... .- attr(*, "dimnames")=List of 2 .. .. ..$: NULL .. .. ..$: chr [1:2] "Green" "Red" ..$ correctedAreas : num [1:246] 3.43 2.94 5.02 3.76 4.08 ... ..$ areacut : num 49 ..$ transformation : chr "log" ..$ image.type : chr [1:3] "BF" "Green" "Red" ..$ dateIndex : chr "ThuMar2416:02:312016" ..$ single.batch.analysis : num 5 ..$ BGmethod : chr "normexp" ..$ maxMix : num 3 ..$ prior.pi : num 0.1 ..$ flex.reps : num 5 ..$ flexmethod : chr "BIC" ..$ RNG : num 999 ..$ GAPgroups : num [1:246, 1:2] 6 4 5 4 4 4 1 4 2 4 ... ..$ clusterFUN : chr "kmeans" ..$ normal.sigma : num 200 ..$ centroids : num [1:6, 1:3] 3 6 5 4 2 ... .- attr(*, "dimnames")=List of 2 .. .. ..$: NULL .. .. ..$: chr [1:3] "Cluster" "Green" "Red" ..$ fixClusters : num 0 ..$ Kmax : num 15 ..$ B.kmeans : num 5 ..$ Path : num [1:6] 3 4 6 2 1 5 ..$ Path.type : chr [1:2] "circular" "clockwise" ..$ UpdatedPath : num [1:6] 1 2 3 4 5 6 ..$ DataSorts : chr [1:246, 1:2] "-0.0524253515714255" "0.0573114934260645" "-0.0658947839764907" "-0.10952104926524" ... .- attr(*, "dimnames")=List of 2 .. .. ..$: NULL .. .. ..$: chr [1:2] "Distance" "Pseudotime" ..$ DDHFupdate : logi FALSE ..$ corrected.VStransformed.exprs: num [1:246, 1:2] 90.2 86.4 88.6 88.1 86.4 ... ..$ VSmethod : chr "DDHFmv" ..$ Progression : num [1:246, 1:2] 110 58 229 64 68 51 215 76 166 85 ... .- attr(*, "dimnames")=List of 2 .. .. ..$: NULL .. .. ..$: chr [1:2] "Pseudotime" "transf.Difference" ..$ Updated.groups : num [1:246] 3 3 6 3 3 3 5 3 4 3 ... ..$ CPs : num [1:5] 22 38 118 169 222 ..$ CPmethod : chr "ECP" ..$ CPsig : num 0.01 ..$ CPgroups : num 5 ..$ CPmingroup : num 15 $ init.path: chr [1:2] "bottom/left" "bottom/left"

Value

cross validation modeling

Index

∗ **datasets**
  clu, 2
  estimates, 7
  estimates.2, 9
  files, 10
  Results, 27
  step1, 32
  step2, 32
  step2.1, 35
  step3, 36
  step3.1, 37
  step4, 37
  steps2_4, 38

clu, 2
cluster2outlier, 4
createFluo, 5

defineLocClusters, 6

estimates, 7
estimates.2, 9

files, 10
Fluo_adjustment, 11
Fluo_CV_modeling, 13
Fluo_CV_prep, 14
Fluo_inspection, 17
Fluo_modeling, 18
Fluo_ordering, 20
FluoSelection_byRun, 21

getFluo, 22
getFluo_byRun, 23

LocationMatrix, 24

pathEstimator, 25

readFiles, 26
Results, 27

simcells, 28
spotEstimator, 29
step1, 32
step2, 32
step2.1, 35
step3, 36
step3.1, 37
step4, 37
steps2_4, 38